Jorgensen

Minh Chau Do

09 10 2023

Daten von Jorgensen

```
#Datensortierung

# In Monate umrechnen
jorgensen_r$V1 <- jorgensen_r$V1 * 12
# In Mortalität umrechnen
jorgensen_r$V2 <- 100 - jorgensen_r$V2

jorgensen_r <- jorgensen_r[complete.cases(jorgensen_r),]

jorgensen_r <- jorgensen_r[order(jorgensen_r$V2), ]

# Werte außerhalb des Bereichs [0, inf] raus
jorgensen_r <- jorgensen_r[jorgensen_r$V1 >= 0, ]

xr_jorgensen <- sort(jorgensen_r$V1)
yr_jorgensen <- jorgensen_r$V2


# In Monate umrechnen
jorgensen_b$V1 <- jorgensen_b$V1 * 12
# In Mortalität umrechnen
jorgensen_b$V2 <- 100 - jorgensen_b$V2

jorgensen_b <- jorgensen_b[complete.cases(jorgensen_b),]

jorgensen_b <- jorgensen_b[order(jorgensen_b$V2), ]

xb_jorgensen <- sort(jorgensen_b$V1)
yb_jorgensen <- jorgensen_b$V2
```

Startfunktion

Um die bestmögliche Anpassung an den vorliegenden Datenpunkten darzustellen wurde mithilfe einer Funktion der beste Startparameter mit dem kleinsten Fehler errmittelt, da die Anpassung stark von den initialen Startwerten abhängig ist.

```
find_best_start_2parameter <- function(p, q, max_beta, max_eta, steps_beta,
                                       steps_eta, fitting_function) {
```

```
  best_errors <- numeric()  # Vektor für Fehlerwerte
  best_starts <- matrix(nrow = 0, ncol = 2)  # Matrix für Startparameter

  for (beta in seq(0, max_beta, by = steps_beta)) {
    for (eta in seq(0, max_eta, by = steps_eta)) {
      start_params <- c(beta, eta)

      # Schätze die Parameter mit den aktuellen Startparametern
      if(identical(as.character(substitute(fitting_function)), "getstuexp2")){
        output <- capture.output({
        result <- getstuexp2(
                        p = p, q = q, start = start_params,
                        show.output = TRUE, plot = FALSE, wert1 = 2
                      )
        })
      }
      else{
        output <- capture.output({
        result <- fitting_function(p = p, q = q, start = start_params,
                              show.output = TRUE, plot = FALSE)
        })
      }

      # Berechne den Fehler (jorgensen_r_1$value) für die aktuellen Startparameter
      current_error <- as.numeric(gsub("\\[1\\]\\s+", "", output[5]))

      # Speichere Fehler und die Startparameter, wenn der Fehler nicht NA ist
      if (!is.na(current_error)) {
        best_errors <- c(best_errors, current_error)
        best_starts <- rbind(best_starts, start_params)
      }
    }
  }


  # Finde den Index des kleinsten Fehlers (ignoriere NA-Werte)
  best_index <- which.min(best_errors)

  # Wähle den besten Startparameter mit dem kleinsten Fehler aus
  best_start <- best_starts[best_index, ]

  # Gib den besten Startparameter und den entsprechenden Fehler aus
  cat("Bester Startparameter:", best_start, "\n")
  cat("Bester Fehler:", best_errors[best_index], "\n")

  return(best_start)
}


find_best_start_3parameter <- function(p, q, max_shape1 = 10, max_shape2 = 10,
                                       max_scale = 10, steps_shape1, steps_shape2,
                                       steps_scale, fitting_function) {
```

```
  best_errors <- numeric()  # Vektor für Fehlerwerte
  best_starts <- matrix(nrow = 0, ncol = 3)  # Matrix für Startparameter

  for (shape1 in seq(0, max_shape1, by = steps_shape1)) {
    for (shape2 in seq(0, max_shape2, by = steps_shape2)) {
      for (scale in seq(0, max_shape1, by = steps_scale)) {
        start_params <- c(shape1, shape2, scale)

        # Schätze die Parameter mit den aktuellen Startparametern
        if(identical(as.character(substitute(fitting_function)), "getstuexp3")){
          output <- capture.output({
            result <- getstuexp3(
                          p = p, q = q, start = start_params,
                          show.output = TRUE, plot = FALSE, wert1 = 2, wert2 = 6)
          })
      }

        else{
          output <- capture.output({
            result <- fitting_function(p = p, q = q, start = start_params,
                                show.output = TRUE, plot = FALSE)
          })
        }
        # Berechne den Fehler (jorgensen_r_1$value) für die aktuellen Startparameter
        current_error <- as.numeric(gsub("\\[1\\]\\s+", "", output[5]))

        # Speichere Fehler und die Startparameter, wenn der Fehler nicht NA ist
        if (!is.na(current_error)) {
          best_errors <- c(best_errors, current_error)
          best_starts <- rbind(best_starts, start_params)
        }
      }
    }
  }

  # Finde den Index des kleinsten Fehlers (ignoriere NA-Werte)
  best_index <- which.min(best_errors)

  # Wähle den besten Startparameter mit dem kleinsten Fehler aus
  best_start <- best_starts[best_index, ]

  # Gib den besten Startparameter und den entsprechenden Fehler aus
  cat("Bester Startparameter:", best_start, "\n")
  cat("Bester Fehler:", best_errors[best_index], "\n")

  return(best_start)
}



find_best_start_4parameter <- function(p, q, max_shape, max_scale, max_rate,
                                       max_mix, steps_shape, steps_scale,
```

```
                                    steps_rate, steps_mix,fitting_function) {

  best_errors <- numeric()  # Vektor für Fehlerwerte
  best_starts <- matrix(nrow = 0, ncol = 4)  # Matrix für Startparameter

  for (shape in seq(0, max_shape, by = steps_shape)) {
    for (scale in seq(0, max_scale, by = steps_scale)) {
      for (rate in seq(0, max_rate, by = steps_rate)) {
        for (mix in seq(0, max_mix, by = steps_mix)) {
          start_params <- c(shape, scale, rate, mix)

          # Schätze die Weibull-Parameter mit den aktuellen Startparametern
          output <- capture.output({
            result <- fitting_function(p = p, q = q, start = start_params,
                                       show.output = TRUE, plot = FALSE)
          })

          # Berechne den Fehler (jorgensen_r_1$value) für die aktuellen Startparameter
          current_error <- as.numeric(gsub("\\[1\\]\\s+", "", output[5]))

          # Speichere Fehler und die Startparameter, wenn der Fehler nicht NA ist
          if (!is.na(current_error)) {
            best_errors <- c(best_errors, current_error)
            best_starts <- rbind(best_starts, start_params)
          }
        }
      }
    }
  }

  # Finde den Index des kleinsten Fehlers (ignoriere NA-Werte)
  best_index <- which.min(best_errors)

  # Wähle den besten Startparameter mit dem kleinsten Fehler aus
  best_start <- best_starts[best_index, ]

  # Gib den besten Startparameter und den entsprechenden Fehler aus
  cat("Bester Startparameter:", best_start, "\n")
  cat("Bester Fehler:", best_errors[best_index], "\n")

  return(best_start)
}
```

Datenanpassung an die Daten von Jorgensen

Weibullverteilung

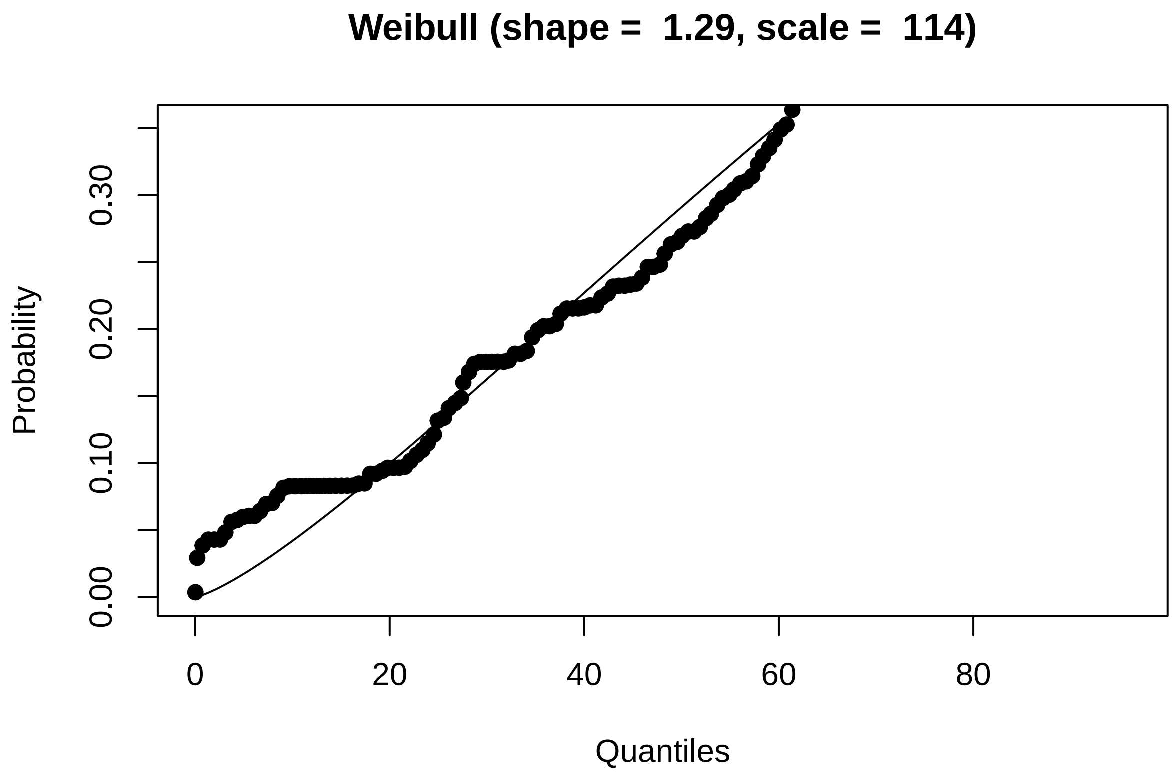
```
# Weibullverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getweibullpar.R")

best_jorgensen_r_1 <- find_best_start_2parameter(p = yr_jorgensen/100, q = xr_jorgensen,
                                                 max_beta = 10, max_eta = 10,
```

```
                                                steps_beta = 1, steps_eta = 1,
                                                fitting_function = getweibullpar)
```

```
## Bester Startparameter: 2 5
## Bester Fehler: 2.18233e-06
```

```
jorgensen_r_1 <- getweibullpar(
                        p = yr_jorgensen/100,
                        q = xr_jorgensen,
                        start = best_jorgensen_r_1,
                        show.output = TRUE,
                        plot = TRUE
                      )
```

```
## $par
## [1]   1.294875 114.000721
##
## $value
## [1] 2.18233e-06
##
## $counts
## function gradient
##       77       77
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

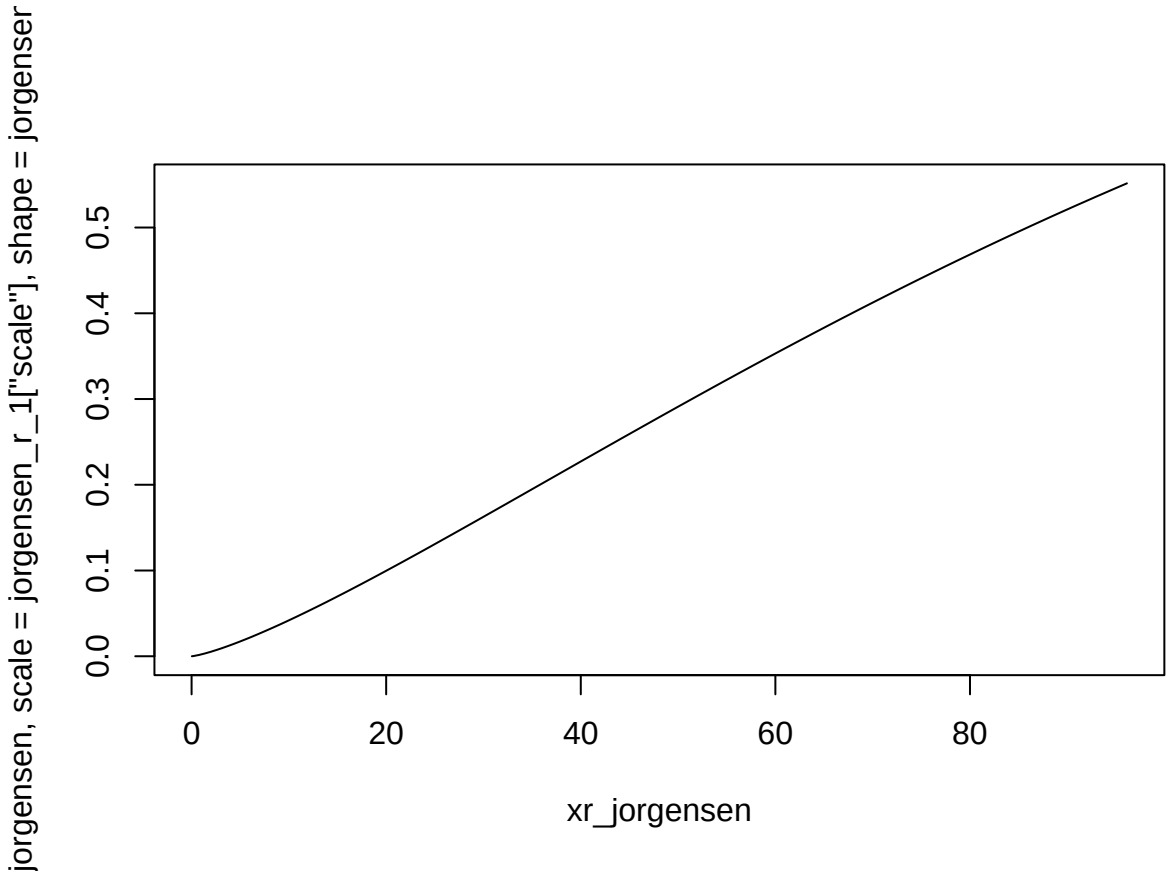


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```r
jorgensen_r_1
```

```
##      shape      scale
##   1.294875 114.000721
```

```r
plot(xr_jorgensen,
     pweibull(xr_jorgensen,
              scale = jorgensen_r_1["scale"],
              shape = jorgensen_r_1["shape"]), type = "l")
```



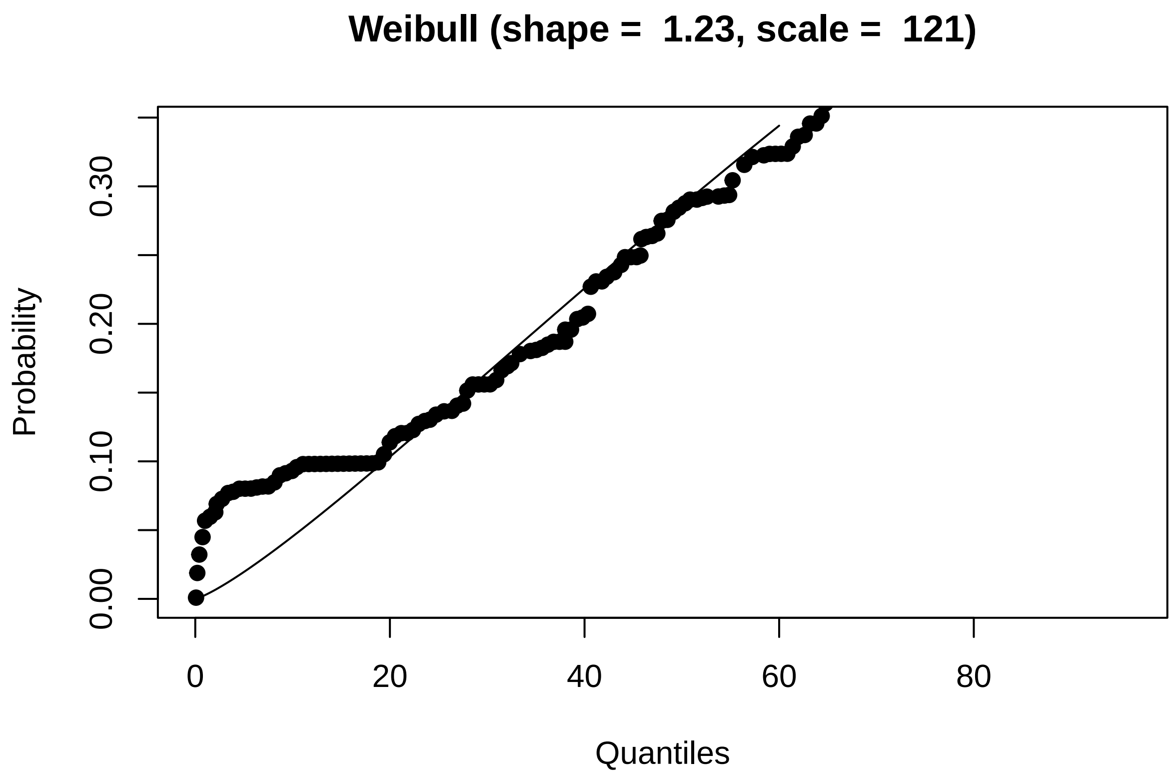

```
best_jorgensen_b_1 <- find_best_start_2parameter(p = yb_jorgensen/100, q = xb_jorgensen,
                                                 max_beta = 10, max_eta = 10,
                                                 steps_beta = 1, steps_eta = 1,
                                                 fitting_function = getweibullpar)
```

```
## Bester Startparameter: 2 2
## Bester Fehler: 3.334577e-06
```

```
jorgensen_b_1 <- getweibullpar(
                        p = yb_jorgensen/100,
                        q = xb_jorgensen,
                        start = best_jorgensen_b_1,
                        show.output = TRUE,
                        plot = TRUE
                       )
```

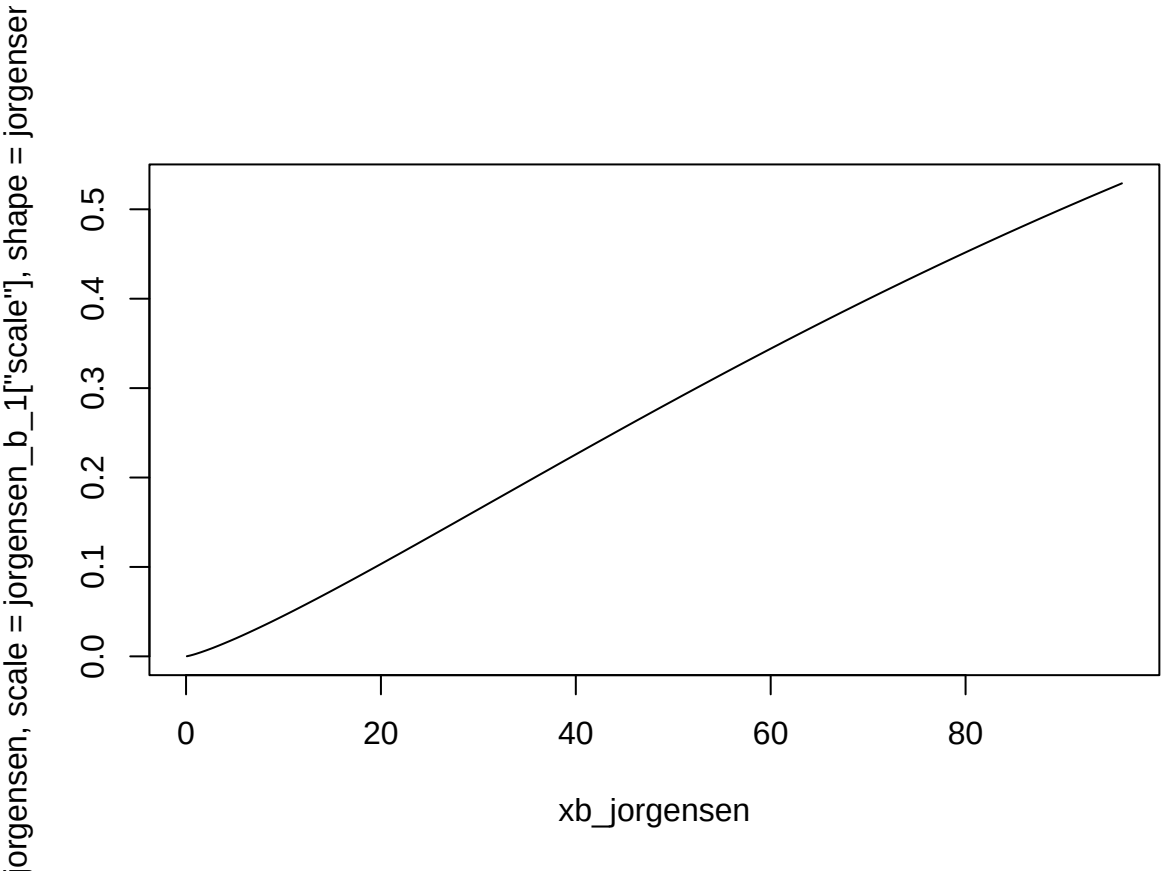
```
## $par
## [1]   1.231547 120.941739
##
## $value
## [1] 3.334577e-06
##
## $counts
## function gradient
##       57       57
##
## $convergence
```

```
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
plot(xb_jorgensen,
     pweibull(xb_jorgensen,
              scale = jorgensen_b_1["scale"],
              shape = jorgensen_b_1["shape"]), type = "l")
```



Exponentiierte Weibullverteilung

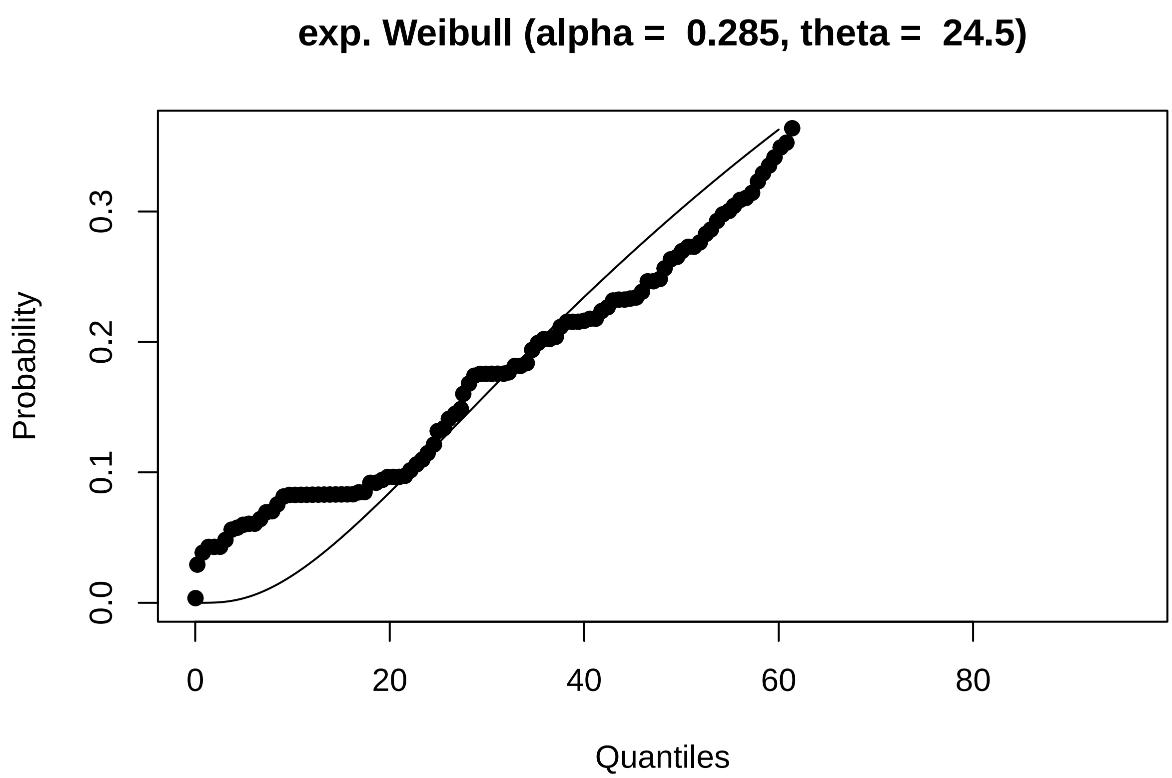
```
# exponentiierte Weibullverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getweibpar.R")

best_weibbull_xr_jorgensen <- find_best_start_2parameter(p = yr_jorgensen/100,
                                                         q = xr_jorgensen,
                                                         max_beta = 10,
                                                         max_eta = 10,
                                                         steps_beta = 1,
                                                         steps_eta = 1,
                                                         fitting_function = getweibpar)
```

```
## Bester Startparameter: 5 0
## Bester Fehler: 4.358661e-06
```

```
weibbull_xr_jorgensen <- getweibpar(
                          p = yr_jorgensen/100,
                          q = xr_jorgensen,
                          start = best_weibbull_xr_jorgensen,
                          show.output = TRUE,
                          plot = TRUE
                        )
```

```
## $par
## [1]  0.2845173 24.5051353
```

```
##
## $value
## [1] 4.358661e-06
##
## $counts
## function gradient
##       38       38
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

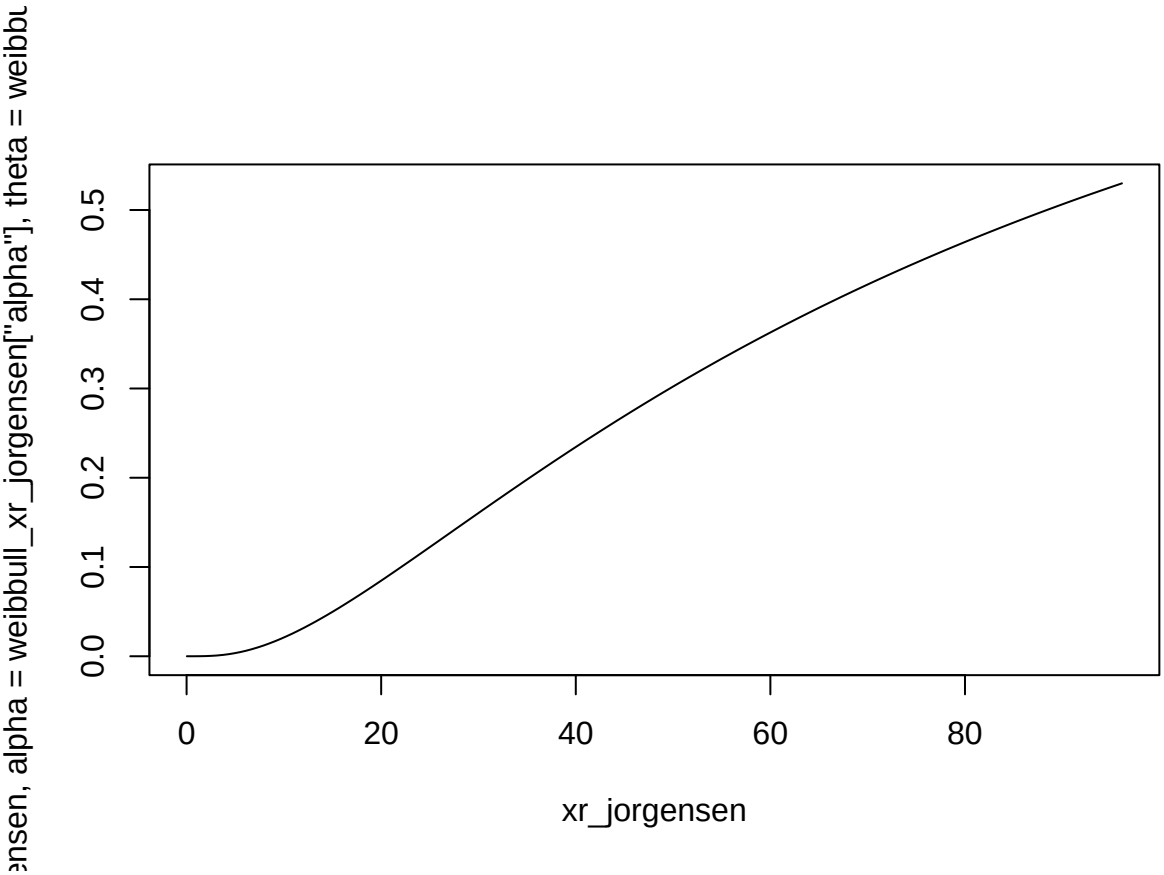


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```r
weibbull_xr_jorgensen
```

```
##      alpha      theta
##  0.2845173 24.5051353
```

```r
plot(xr_jorgensen,
     pexpo.weibull(xr_jorgensen,
                   alpha = weibbull_xr_jorgensen ["alpha"],
                   theta = weibbull_xr_jorgensen ["theta"]), type = "l")
```



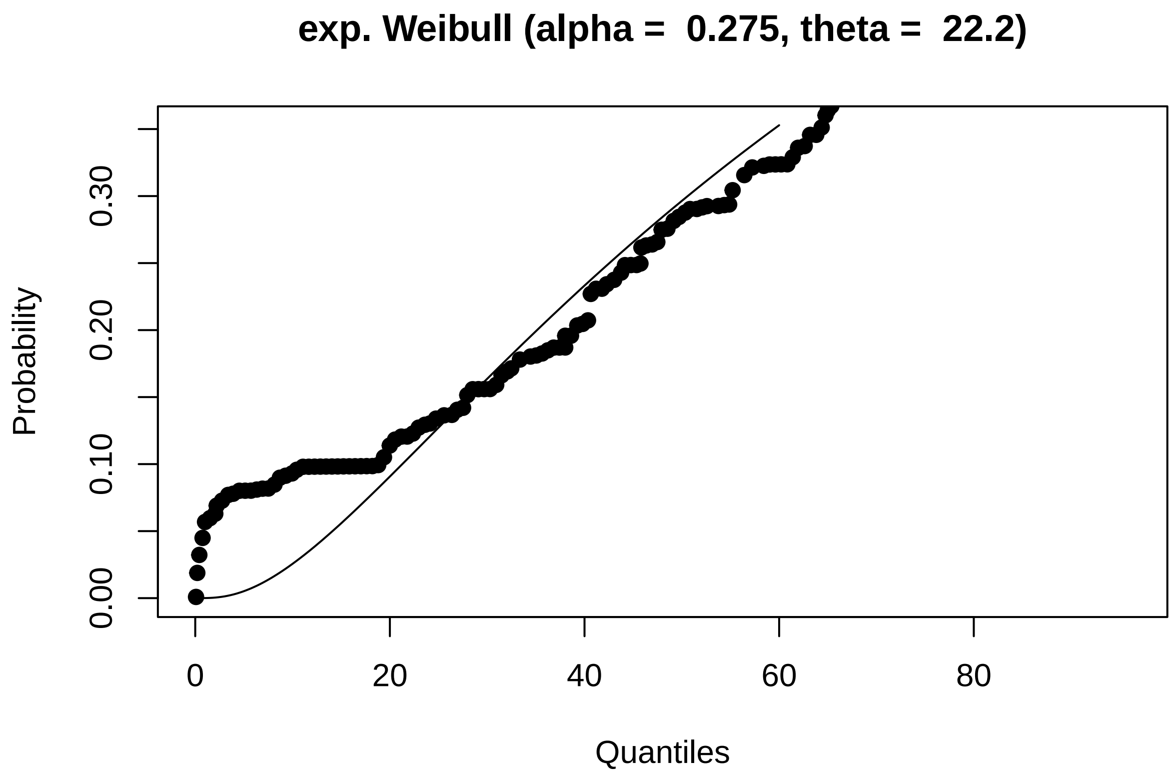

```
best_weibbull_xb_jorgensen <- find_best_start_2parameter(p = yb_jorgensen/100,
                                                         q = xb_jorgensen,
                                                         max_beta = 10,
                                                         max_eta = 10,
                                                         steps_beta = 1,
                                                         steps_eta = 1,
                                                         fitting_function = getweibpar)
```

```
## Bester Startparameter: 0 4
## Bester Fehler: 6.085128e-06
```

```
weibbull_xb_jorgensen  <- getweibpar(
                          p = yb_jorgensen/100,
                          q = xb_jorgensen,
                          start = best_weibbull_xb_jorgensen,
                          show.output = TRUE,
                          plot = TRUE
                         )
```

```
## $par
## [1]  0.2750718 22.2315848
##
## $value
## [1] 6.085128e-06
##
## $counts
## function gradient
```

```
##       28       28
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

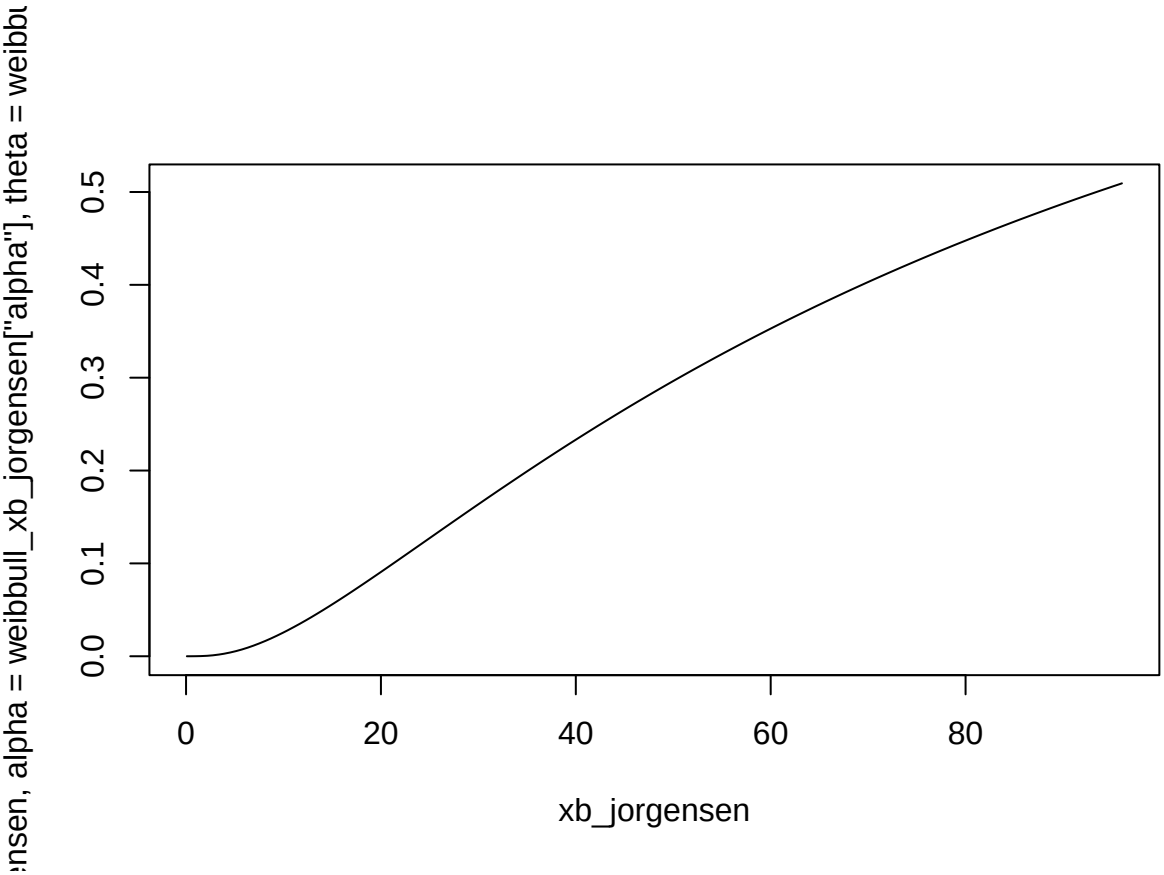


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```r
weibbull_xb_jorgensen
```

```
##      alpha      theta
##  0.2750718 22.2315848
```

```r
plot(xb_jorgensen,
     pexpo.weibull(xb_jorgensen,
                  alpha = weibbull_xb_jorgensen ["alpha"],
                  theta = weibbull_xb_jorgensen ["theta"]), type = "l")
```



Mischung aus Weibull- und Exponentialverteilung

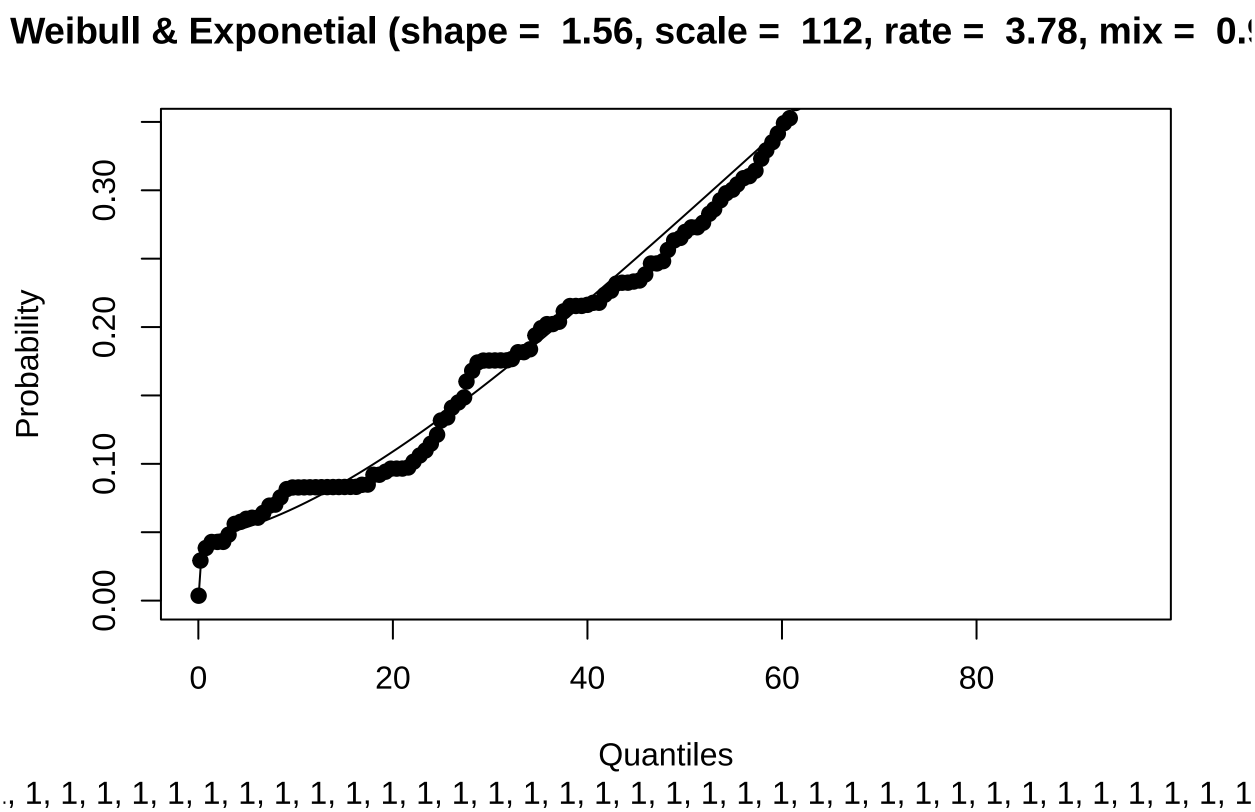
```
# Mischung aus Weibull- und Exponentialverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getweibex.R")

best_weibex_xr_jorgensen <- find_best_start_4parameter(p = yr_jorgensen/100,
                                                       q = xr_jorgensen,
                                                       max_shape = 1,
                                                       max_scale = 100,
                                                       max_rate = 0.7,
                                                       max_mix = 1,
                                                       steps_shape = 0.1,
                                                       steps_scale = 20,
                                                       steps_rate = 0.1,
                                                       steps_mix = 0.1,
                                                       fitting_function = getweibex)
```

```
## Bester Startparameter: 0.5 100 0.6 0.2
## Bester Fehler: 9.427333e-07
```

```
weibex_xr_jorgensen <- getweibex(
                        p = yr_jorgensen/100,
                        q = xr_jorgensen,
                        start = best_weibex_xr_jorgensen, # c(0.5, 60, 0.4, 0.1),
                        show.output = TRUE,
                        plot = TRUE
```

```
                        )
```

```
## $par
## [1]   1.558657 112.219015   3.780190   3.025024
##
## $value
## [1] 9.427333e-07
##
## $counts
## function gradient
##       22       22
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

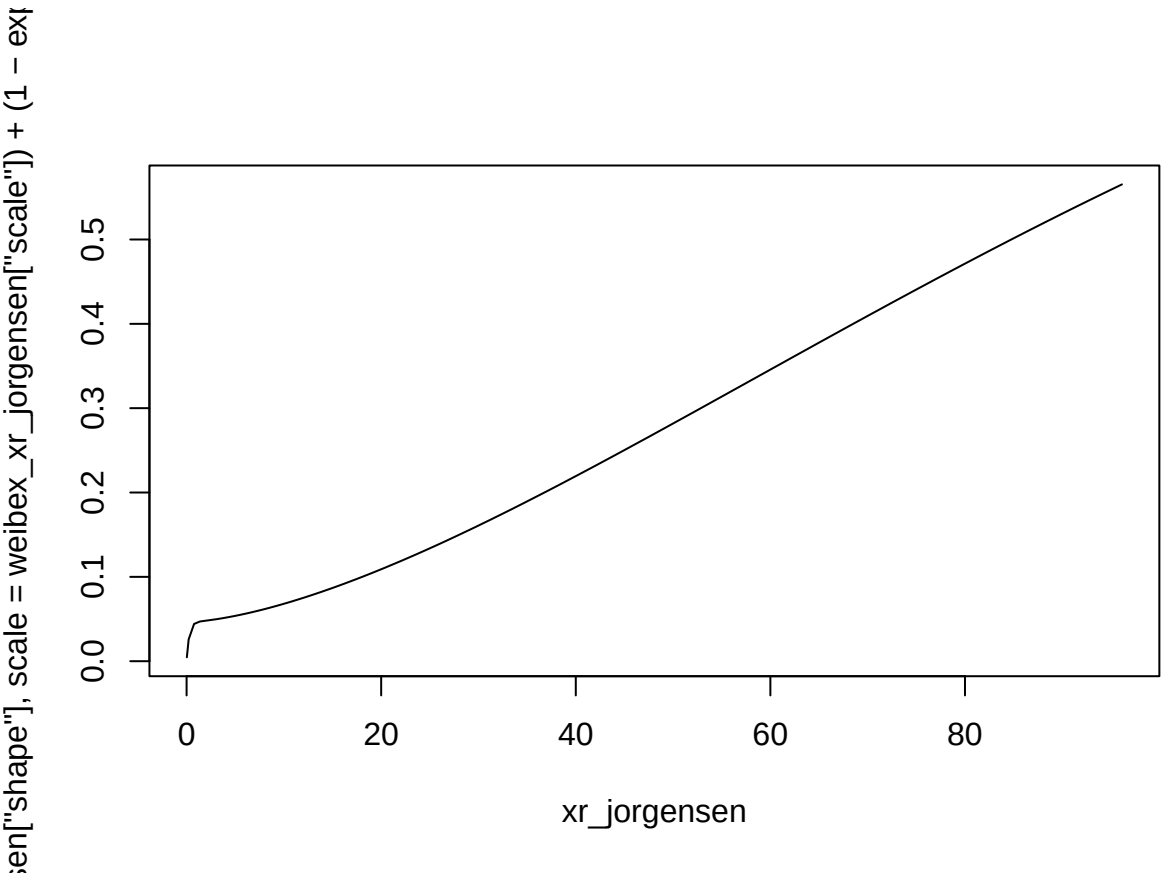



```
weibex_xr_jorgensen
```

```
##      shape      scale       rate        mix
##   1.558657 112.219015   3.780190   3.025024
```

```
# Wenn Ergebnisse aus weibex_xr1_1 von Funktion abgelesen
plot(xr_jorgensen,
     (exp(weibex_xr_jorgensen["mix"]) / ( 1 + exp(weibex_xr_jorgensen["mix"])) *
        stats::pweibull(q = xr_jorgensen,
```

```
                        shape = weibex_xr_jorgensen["shape"],
                        scale = weibex_xr_jorgensen["scale"]) +
       (1 - exp(weibex_xr_jorgensen["mix"]) / ( 1 + exp(weibex_xr_jorgensen["mix"]))) *
       stats::pexp(q = xr_jorgensen,
                   rate = weibex_xr_jorgensen["rate"])),
     type = "l")
```



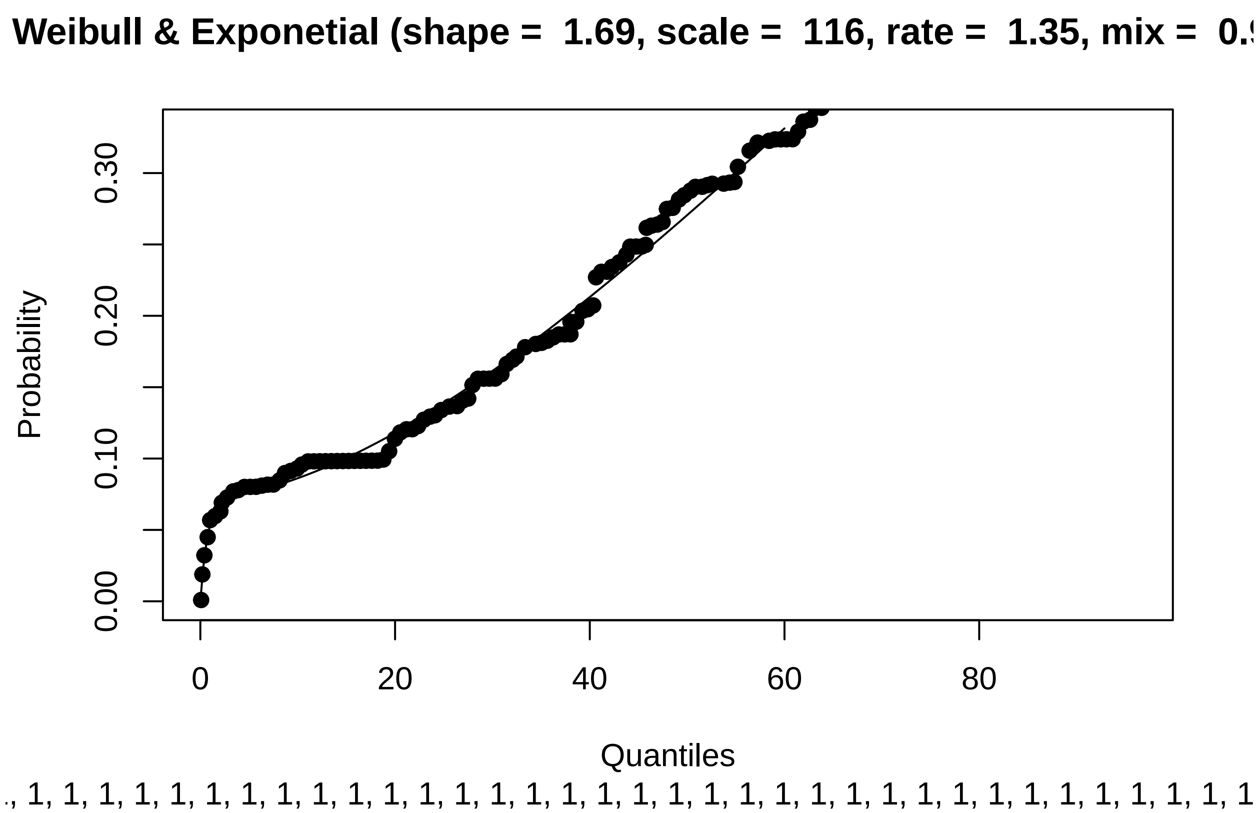

```
best_weibex_xb_jorgensen <- find_best_start_4parameter(p = yb_jorgensen/100,
                                                       q = xb_jorgensen,
                                                       max_shape = 1,
                                                       max_scale = 100,
                                                       max_rate = 0.7,
                                                       max_mix = 1,
                                                       steps_shape = 0.1,
                                                       steps_scale = 20,
                                                       steps_rate = 0.1,
                                                       steps_mix = 0.1,
                                                       fitting_function = getweibex)
```

```
## Bester Startparameter: 0.8 100 0.7 0.8
## Bester Fehler: 5.537916e-07
```

```
weibex_xb_jorgensen <- getweibex(
                        p = yb_jorgensen/100,
                        q = xb_jorgensen,
                        start = best_weibex_xb_jorgensen, # c(0.5, 60, 0.7, 0.1),
                        show.output = TRUE,
```

```r
                         plot = TRUE
                       )
```

```
## $par
## [1]   1.688798 116.055116   1.347347   2.563400
##
## $value
## [1] 5.537916e-07
##
## $counts
## function gradient
##       24       24
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

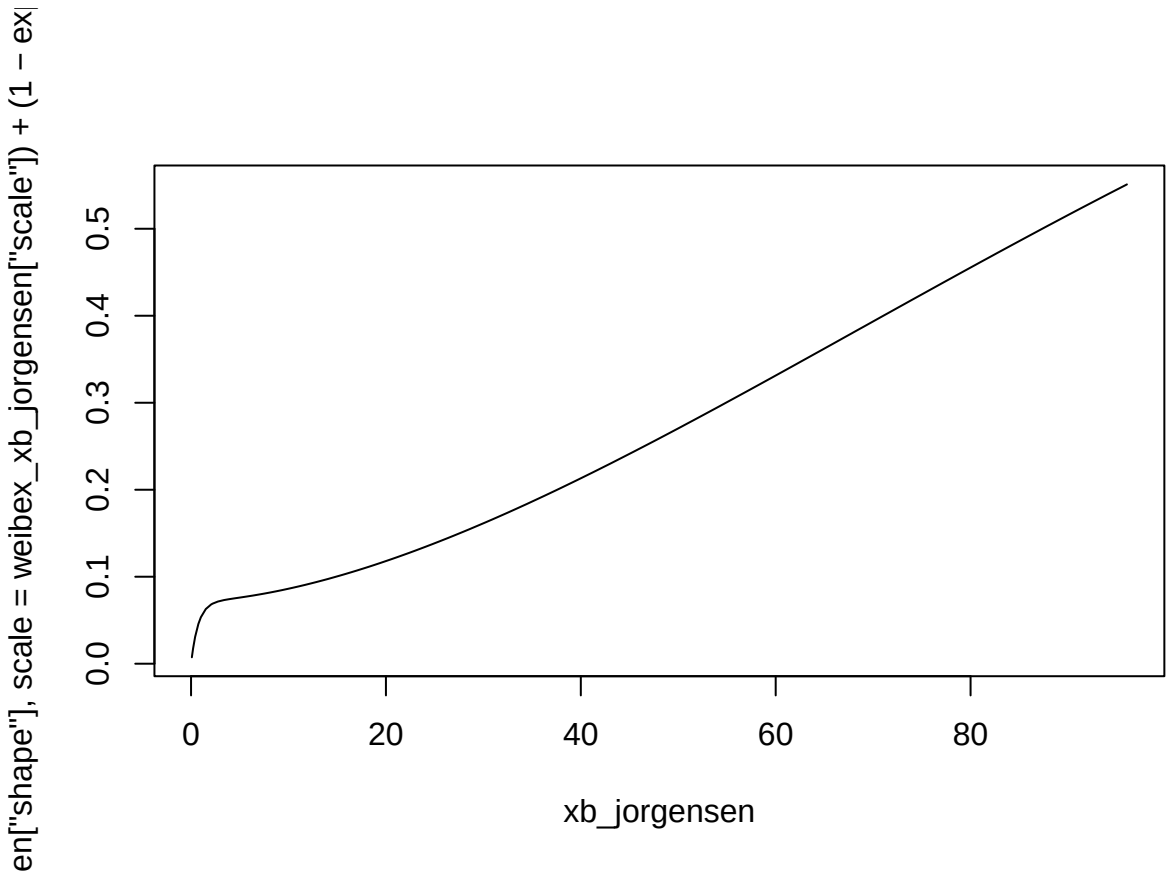



```r
weibex_xb_jorgensen
```

```
##      shape      scale       rate        mix
##   1.688798 116.055116   1.347347   2.563400
```

```r
plot(xb_jorgensen,
     (exp(weibex_xb_jorgensen["mix"]) / ( 1 + exp(weibex_xb_jorgensen["mix"])) *
        stats::pweibull(q = xb_jorgensen,
```

```
                        shape = weibex_xb_jorgensen["shape"],
                        scale = weibex_xb_jorgensen["scale"]) +
       (1 - exp(weibex_xb_jorgensen["mix"]) / ( 1 + exp(weibex_xb_jorgensen["mix"]))) *
       stats::pexp(q = xb_jorgensen,
                   rate = weibex_xb_jorgensen["rate"])),
     type = "l")
```



Mischung von exponentiierter Weibull- und Exponentialverteilung

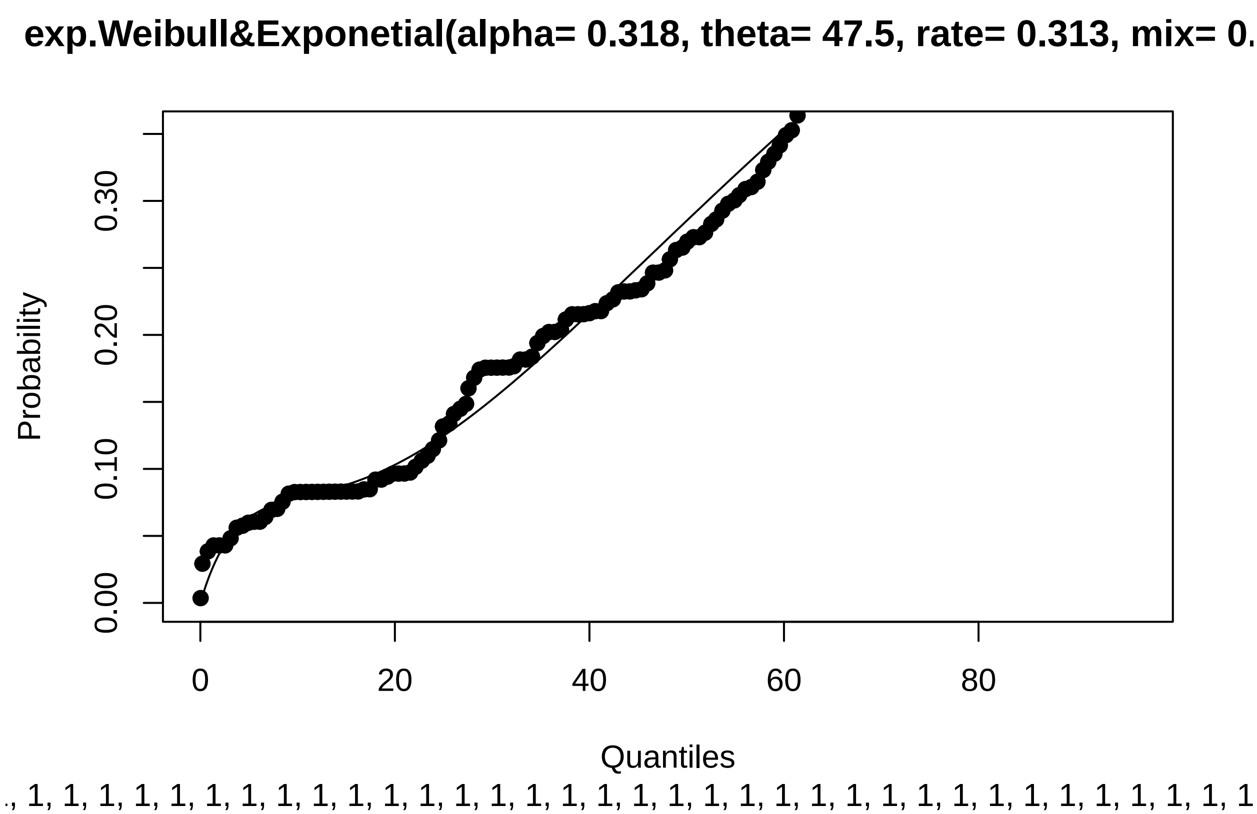
```
# Mischung von exponentiierter Weibull- und Exponentialverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/get2weibex.R")

best_weibex2_xr_jorgensen <- find_best_start_4parameter(p = yr_jorgensen/100,
                                                        q = xr_jorgensen,
                                                        max_shape = 1,
                                                        max_scale = 100,
                                                        max_rate = 0.7,
                                                        max_mix = 1,
                                                        steps_shape = 0.1,
                                                        steps_scale = 20,
                                                        steps_rate = 0.1,
                                                        steps_mix = 0.1,
                                                        fitting_function = get2weibex)
```

```
## Bester Startparameter: 0.1 0 0.4 0.4
```

```
## Bester Fehler: 8.776434e-07
```

```r
weibex2_xr_jorgensen <- get2weibex(
                          p = yr_jorgensen/100,
                          q = xr_jorgensen,
                          start = best_weibex2_xr_jorgensen,
                          show.output = TRUE,
                          plot = TRUE
                        )
```

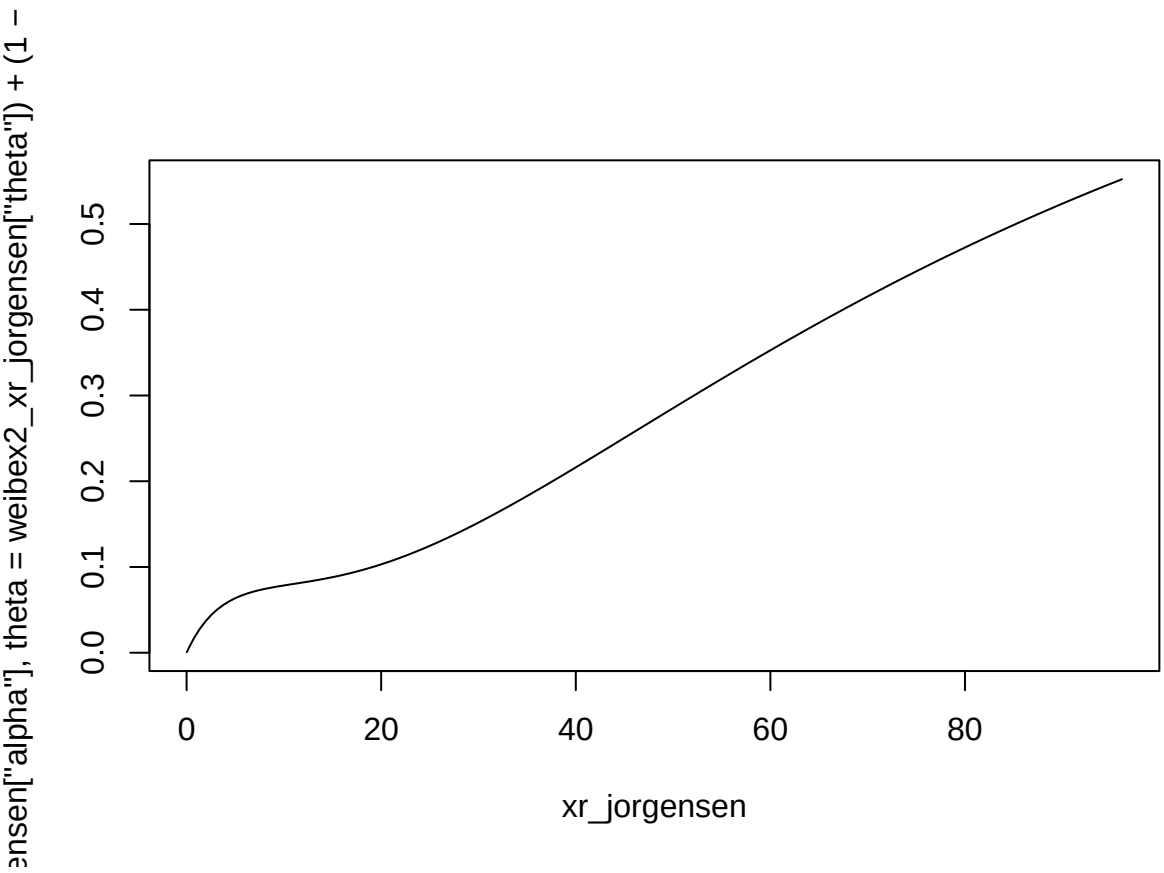
```
## $par
## [1]  0.3180856 47.5324993  0.3130429  2.4378076
##
## $value
## [1] 8.776434e-07
##
## $counts
## function gradient
##       71       71
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```
weibex2_xr_jorgensen
```

```
##      alpha      theta       rate        mix
##  0.3180856 47.5324993  0.3130429  2.4378076
```

```
plot(xr_jorgensen,
     (exp(weibex2_xr_jorgensen["mix"]) / ( 1 + exp(weibex2_xr_jorgensen["mix"])) *
        reliaR::pexpo.weibull(q = xr_jorgensen,
                              alpha = weibex2_xr_jorgensen["alpha"],
                              theta = weibex2_xr_jorgensen["theta"]) +
        (1 - exp(weibex2_xr_jorgensen["mix"]) / ( 1 + exp(weibex2_xr_jorgensen["mix"]))) *
        stats::pexp(q = xr_jorgensen,
                    rate = weibex2_xr_jorgensen["rate"])),
     type = "l")
```

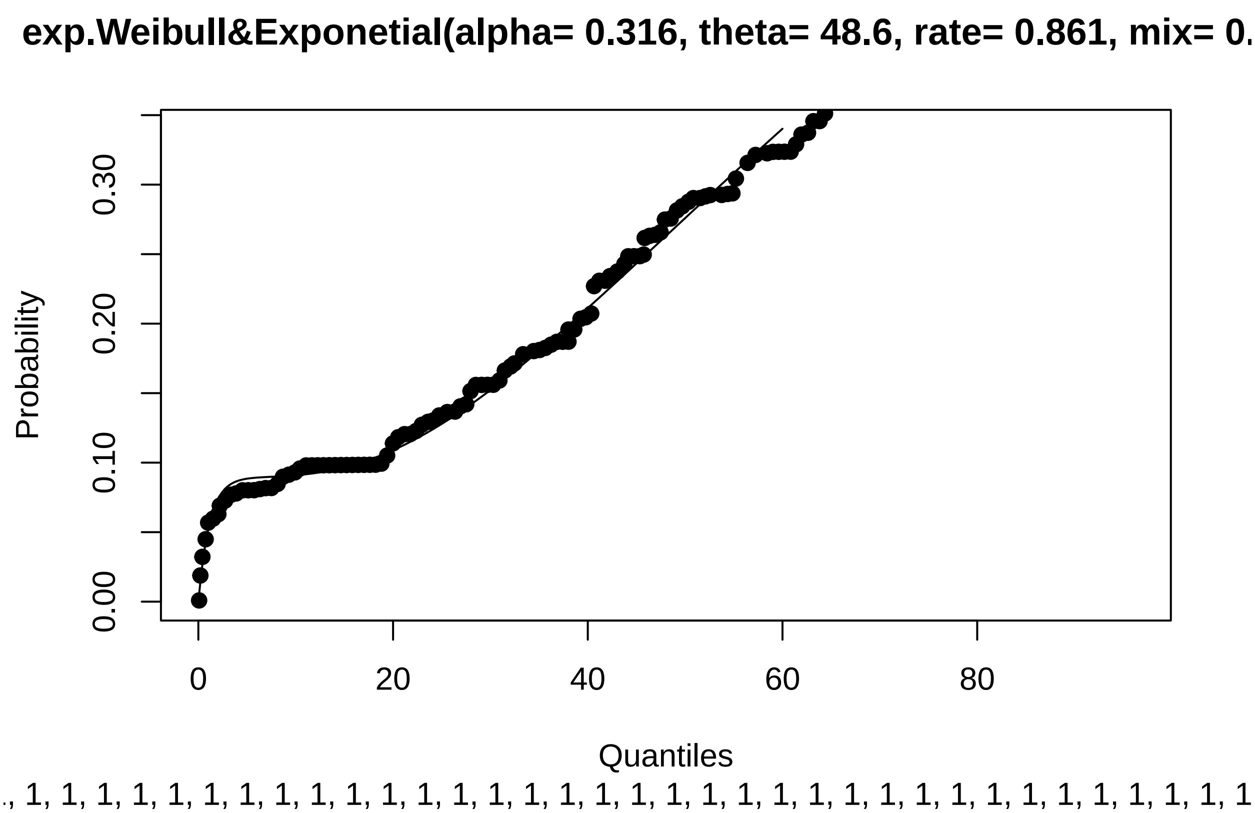



```
best_weibex2_xb_jorgensen <- find_best_start_4parameter(p = yb_jorgensen/100,
                                                        q = xb_jorgensen,
                                                        max_shape = 1,
                                                        max_scale = 100,
                                                        max_rate = 0.7,
                                                        max_mix = 1,
                                                        steps_shape = 0.1,
                                                        steps_scale = 20,
                                                        steps_rate = 0.1,
                                                        steps_mix = 0.1,
                                                        fitting_function = get2weibex)
```

```
## Bester Startparameter: 0 0 0.4 0.6
## Bester Fehler: 7.774687e-07
```

```
weibex2_xb_jorgensen <- get2weibex(
                            p = yb_jorgensen/100,
                            q = xb_jorgensen,
                            start = best_weibex2_xb_jorgensen,
                            show.output = TRUE,
                            plot = TRUE
                          )
```

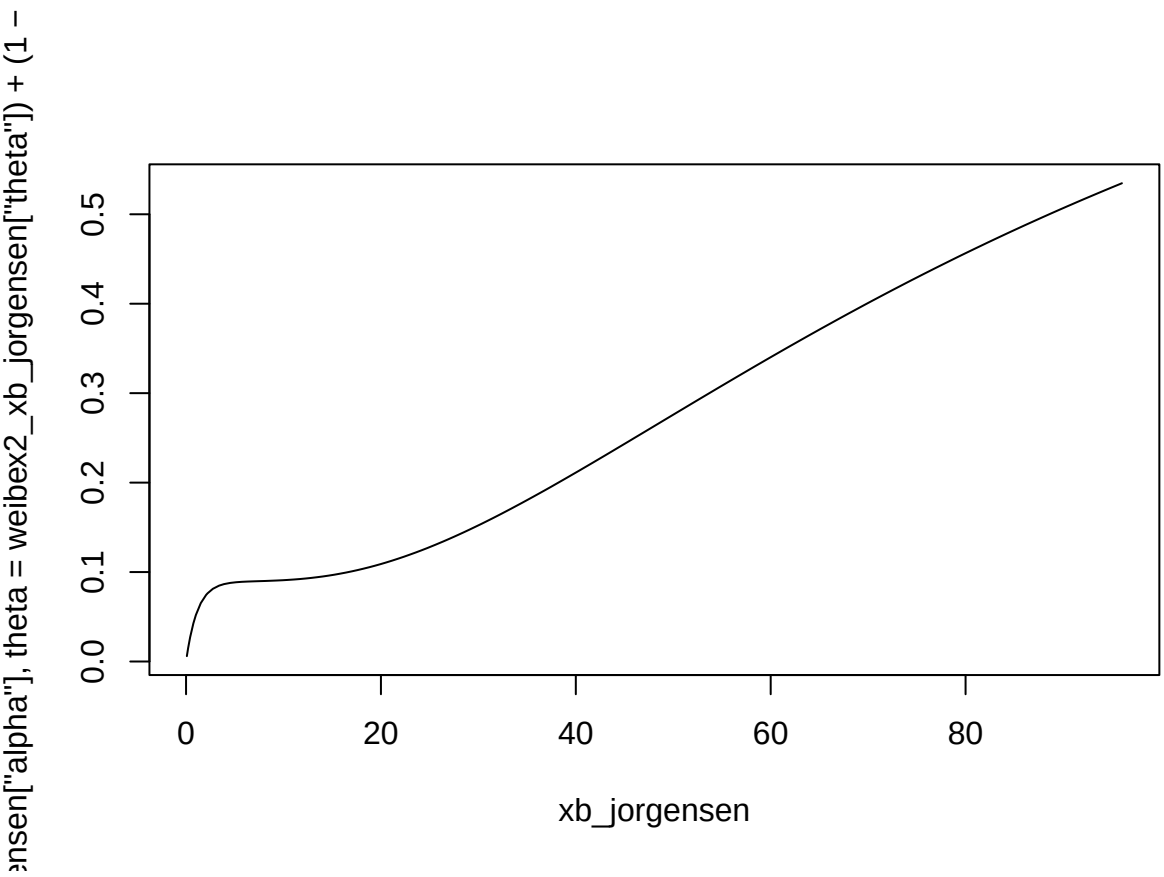
```
## $par
## [1]  0.3157402 48.6334591  0.8606514  2.3178641
##
## $value
## [1] 7.774687e-07
##
## $counts
## function gradient
##       48       48
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



```
weibex2_xb_jorgensen
```

```
##      alpha      theta       rate        mix
##  0.3157402 48.6334591  0.8606514  2.3178641
```

```
plot(xb_jorgensen,
     (exp(weibex2_xb_jorgensen["mix"]) / ( 1 + exp(weibex2_xb_jorgensen["mix"])) *
        reliaR::pexpo.weibull(q = xb_jorgensen,
                              alpha = weibex2_xb_jorgensen["alpha"],
                              theta = weibex2_xb_jorgensen["theta"]) +
        (1 - exp(weibex2_xb_jorgensen["mix"]) / ( 1 + exp(weibex2_xb_jorgensen["mix"]))) *
        stats::pexp(q = xb_jorgensen,
                    rate = weibex2_xb_jorgensen["rate"])),
     type = "l")
```



Exponentiierte Weibullverteilung ohne Lambda = 1

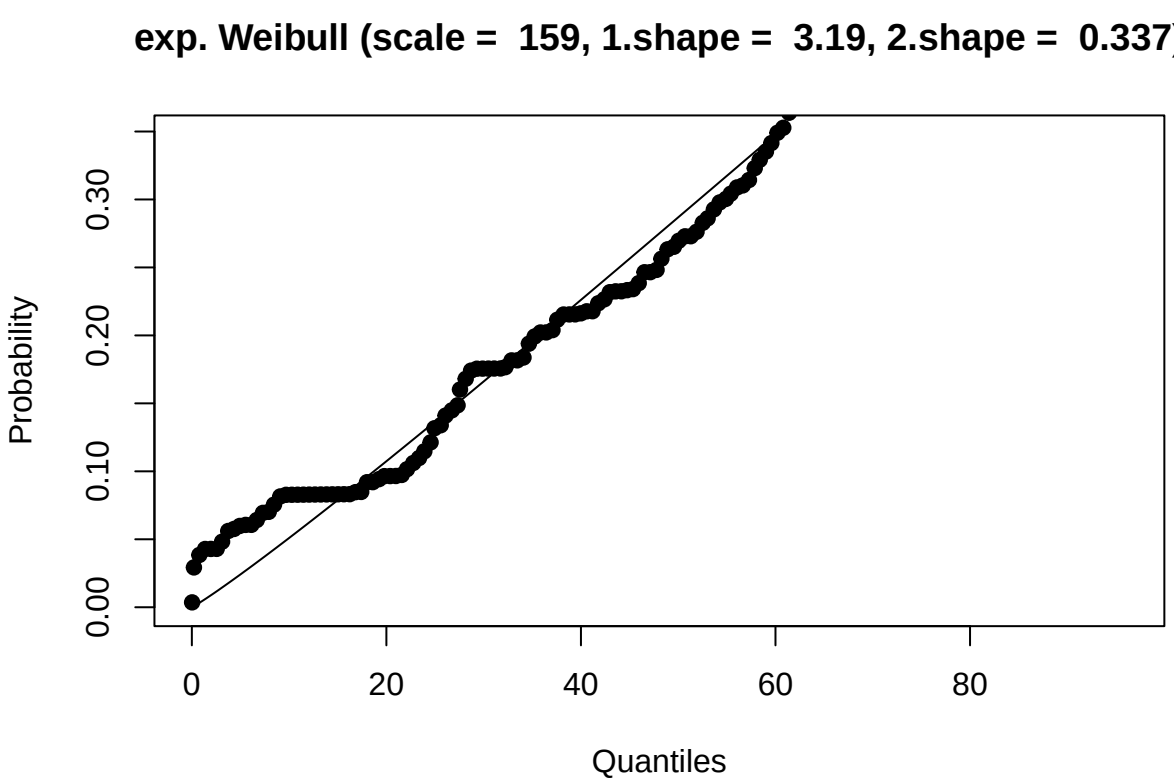
```
# exponentiierte Weibullverteilung ohne Lambda = 1
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getexpweib.R")

best_expweib_xr_jorgensen <- find_best_start_3parameter(p = yr_jorgensen/100,
                                                        q = xr_jorgensen,
                                                        max_shape1 = 10,
                                                        max_shape2 = 10,
                                                        max_scale = 10,
                                                        steps_shape1 = 1,
```

```
                                                    steps_shape2 = 1,
                                                    steps_scale = 1,
                                                    fitting_function = getexpweib)
```

```
## Bester Startparameter: 9 10 7
## Bester Fehler: 1.799463e-06
```

```
expweib_xr_jorgensen <- getexpweib(
                          p = yr_jorgensen/100,
                          q = xr_jorgensen,
                          start = best_expweib_xr_jorgensen,
                          show.output = TRUE,
                          plot = TRUE
                        )
```

```
## $par
## [1] 158.9017624   3.1898719   0.3374211
##
## $value
## [1] 1.799463e-06
##
## $counts
## function gradient
##       46       46
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

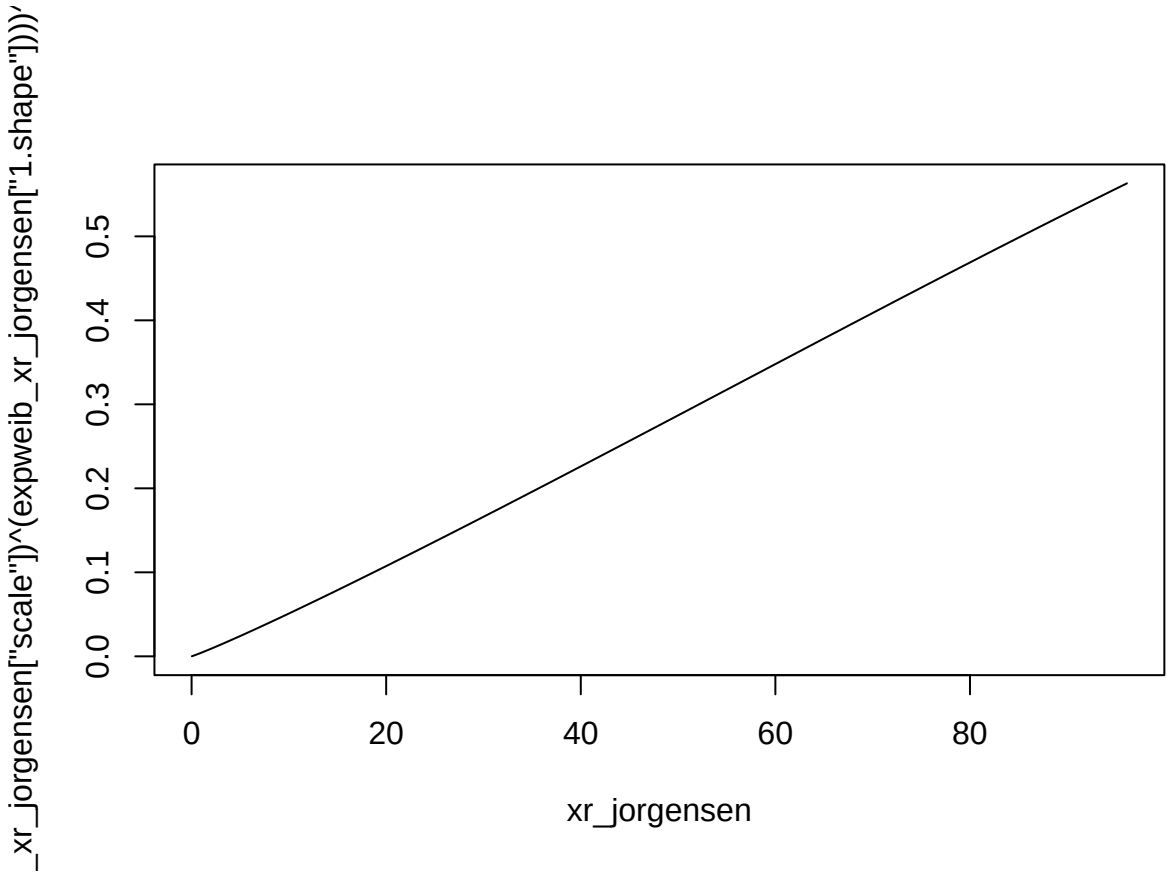


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
expweib_xr_jorgensen
```

```
##       scale     1.shape     2.shape
## 158.9017624   3.1898719   0.3374211
```

```
plot(xr_jorgensen,
     (1 - exp(-(xr_jorgensen / expweib_xr_jorgensen["scale"])^
                      (expweib_xr_jorgensen["1.shape"]))) ^ (expweib_xr_jorgensen["2.shape"]),
     type = "l")
```



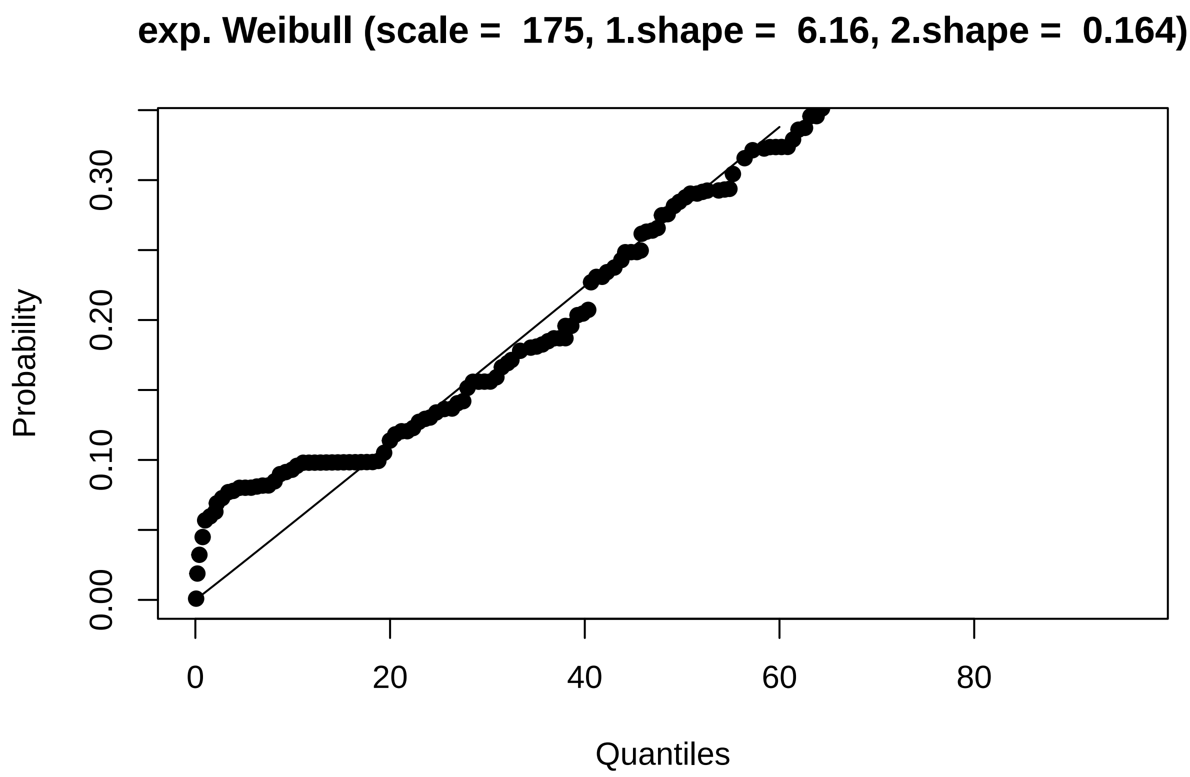

```
best_expweib_xb_jorgensen <- find_best_start_3parameter(p = yb_jorgensen/100,
                                                        q = xb_jorgensen,
                                                        max_shape1 = 10,
                                                        max_shape2 = 10,
                                                        max_scale = 10,
                                                        steps_shape1 = 1,
                                                        steps_shape2 = 1,
                                                        steps_scale = 1,
                                                        fitting_function = getexpweib)
```

```
## Bester Startparameter: 5 8 3
## Bester Fehler: 2.349781e-06
```

```
expweib_xb_jorgensen <- getexpweib(
                          p = yb_jorgensen/100,
                          q = xb_jorgensen,
                          start = best_expweib_xb_jorgensen,
                          show.output = TRUE,
                          plot = TRUE
                        )
```

```
## $par
## [1] 175.1217978   6.1644571   0.1642775
##
## $value
## [1] 2.349781e-06
##
```

```
## $counts
## function gradient
##       57       57
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

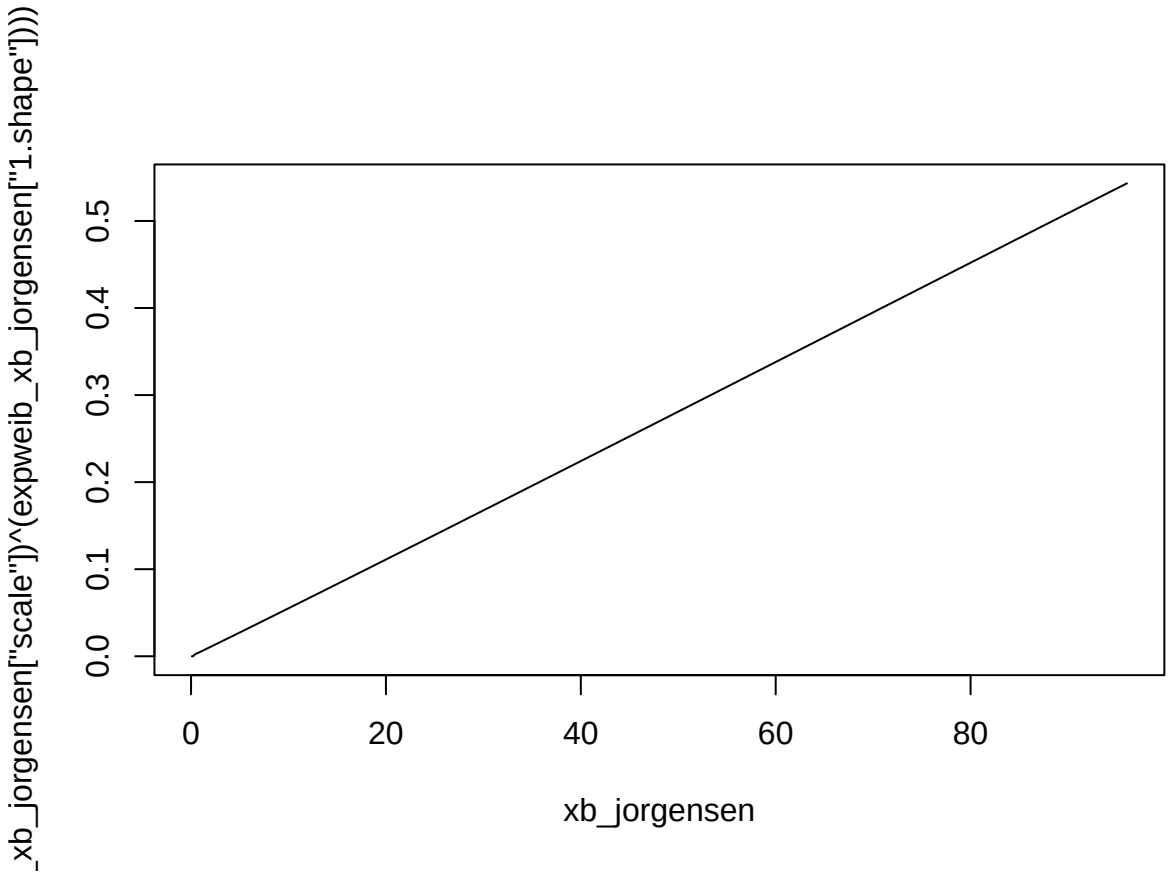


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
expweib_xb_jorgensen
```

```
##       scale     1.shape     2.shape
## 175.1217978   6.1644571   0.1642775
```

```
plot(xb_jorgensen,
     (1 - exp(-(xb_jorgensen / expweib_xb_jorgensen["scale"])^(expweib_xb_jorgensen["1.shape"])))^(expwe
     type = "l")
```



3-Stufige Exponetialverteilung

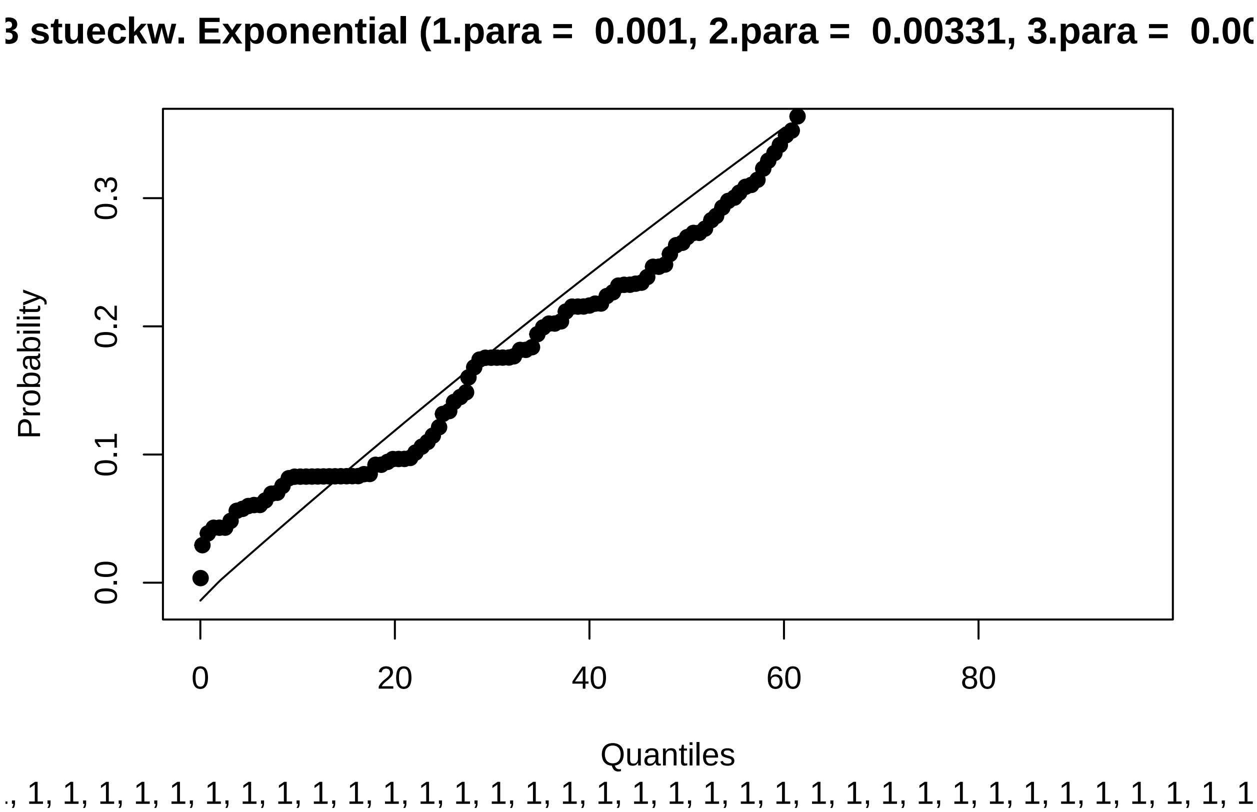
```
# 3-Stufige Exponetialverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getstuexp3.R")

best_stuexp3_xr_jorgensen <- find_best_start_3parameter(p = yr_jorgensen/100,
                                                        q = xr_jorgensen,
                                                        max_shape1 = 0.1,
                                                        max_shape2 = 0.1,
                                                        max_scale = 0.1,
                                                        steps_shape1 = 0.01,
                                                        steps_shape2 = 0.01,
                                                        steps_scale = 0.01,
                                                        fitting_function = getstuexp3)
```

```
## Bester Startparameter: 0.01 0.02 0.02
## Bester Fehler: 3.111139e-06
```

```
stuexp3_xr_jorgensen <- getstuexp3(
                        p = yr_jorgensen/100,
                        q = xr_jorgensen,
                        start = best_stuexp3_xr_jorgensen,
                        show.output = TRUE,
                        plot = TRUE,
                        wert1 = 2,
                        wert2 = 6
```

```
                      )
```

```
## $par
## [1] 0.001000000 0.003313931 0.003442640
##
## $value
## [1] 3.111139e-06
##
## $counts
## function gradient
##       14       14
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

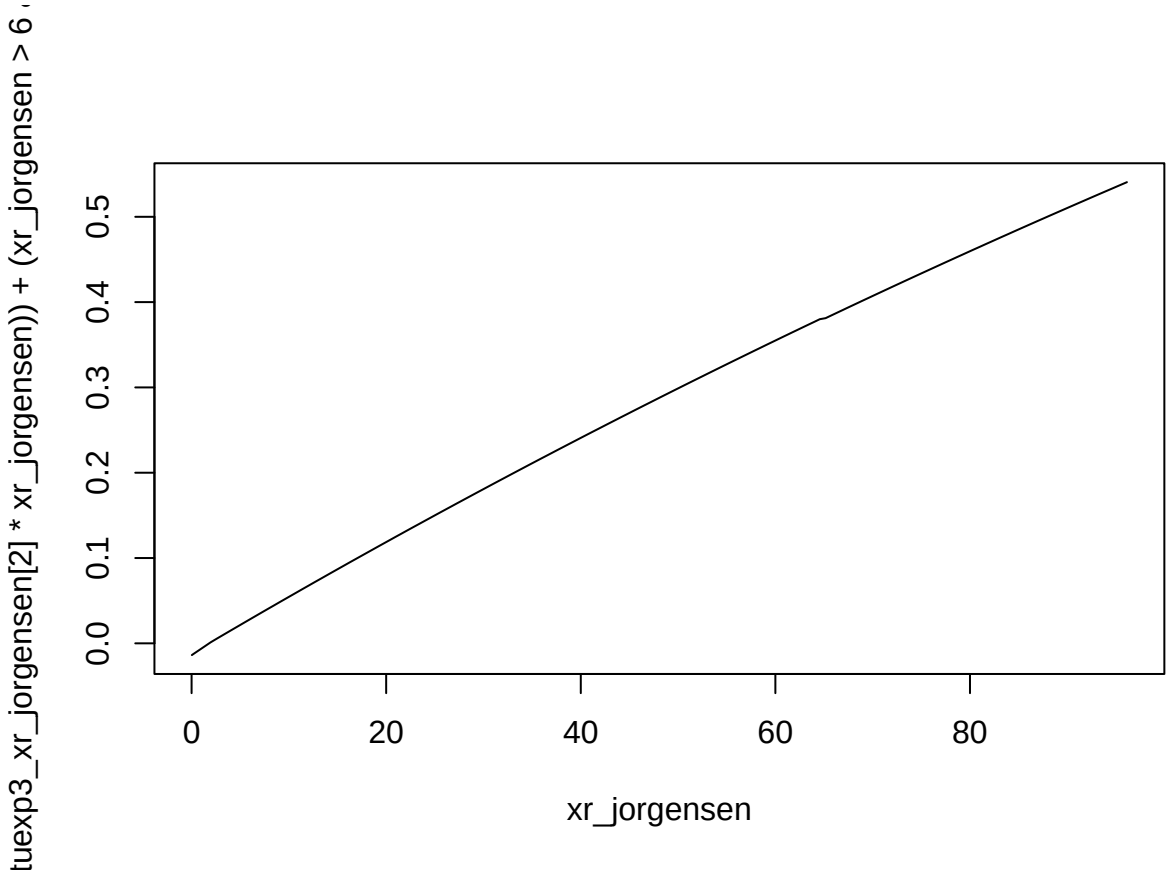



```
stuexp3_xr_jorgensen
```

```
##       1.para       2.para       3.para
## 0.001000000 0.003313931 0.003442640
```

```
plot(xr_jorgensen,
     ((xr_jorgensen > 0 & xr_jorgensen <= 2 ) * (1 - exp(-stuexp3_xr_jorgensen[1] * xr_jorgensen)) +
                              (xr_jorgensen > 2 & xr_jorgensen <= 6 ) *
              (1 - exp(-stuexp3_xr_jorgensen[1] * 2))  +
```

```
                                    (exp(-2 * stuexp3_xr_jorgensen[2]) -
                                            exp(-stuexp3_xr_jorgensen[2] * xr_jorgensen)) +
                                    (xr_jorgensen > 6 & xr_jorgensen <= 65 ) *
            (1 - exp(-stuexp3_xr_jorgensen[1] * 2)) +
                                    (exp(-2 * stuexp3_xr_jorgensen[2]) - exp(-6 * stuexp3_xr_jorgensen[2
                                    (exp(-6 * stuexp3_xr_jorgensen[3]) -
                                            exp(-stuexp3_xr_jorgensen[3] * xr_jorgensen))),
    type = "l")
```



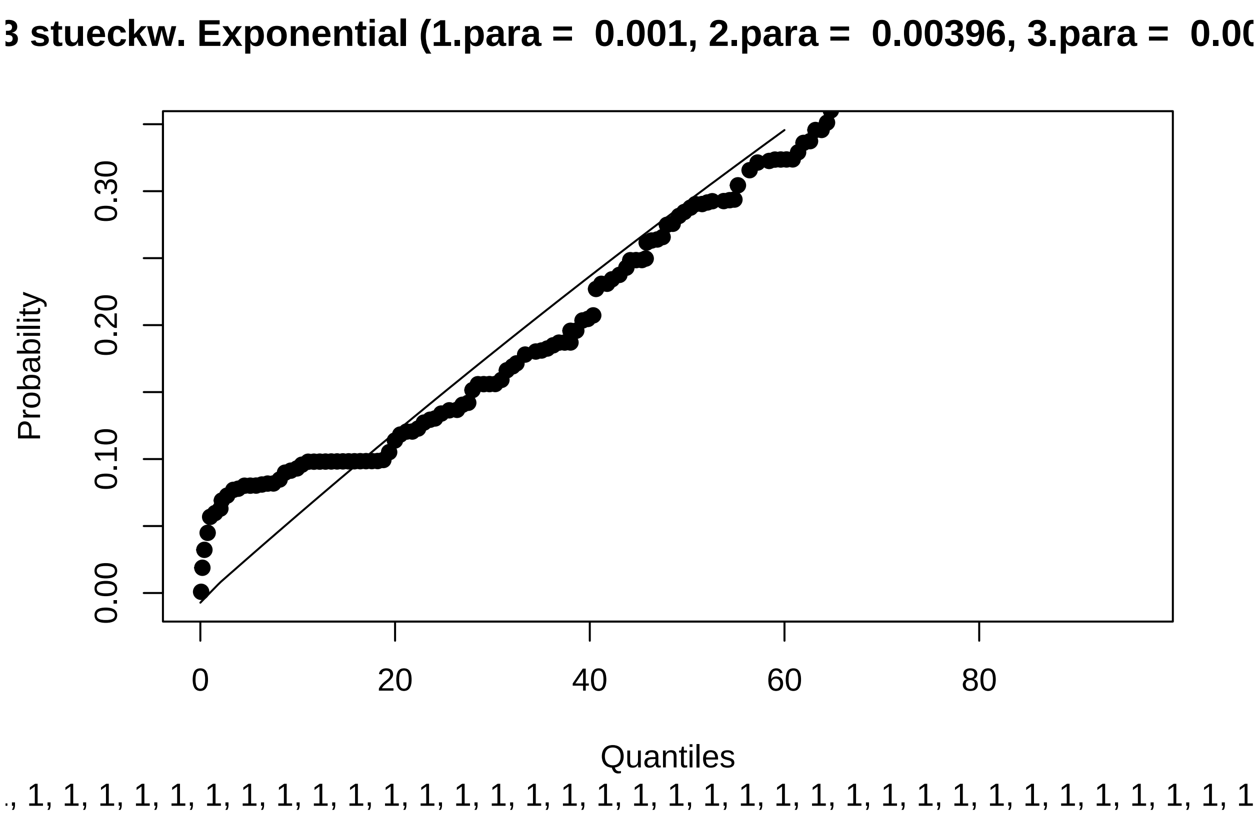

```
best_stuexp3_xb_jorgensen <- find_best_start_3parameter(p = yb_jorgensen/100,
                                                        q = xb_jorgensen,
                                                        max_shape1 = 0.1,
                                                        max_shape2 = 0.1,
                                                        max_scale = 0.1,
                                                        steps_shape1 = 0.01,
                                                        steps_shape2 = 0.01,
                                                        steps_scale = 0.01,
                                                        fitting_function = getstuexp3)
```

```
## Bester Startparameter: 0 0.01 0.01
## Bester Fehler: 3.478313e-06
```

```
stuexp3_xb_jorgensen <- getstuexp3(
                        p = yb_jorgensen/100,
                        q = xb_jorgensen,
                        start = best_stuexp3_xb_jorgensen,
                        show.output = TRUE,
```

```r
                          plot = TRUE,
                          wert1 = 2,
                          wert2 = 6
                        )
```

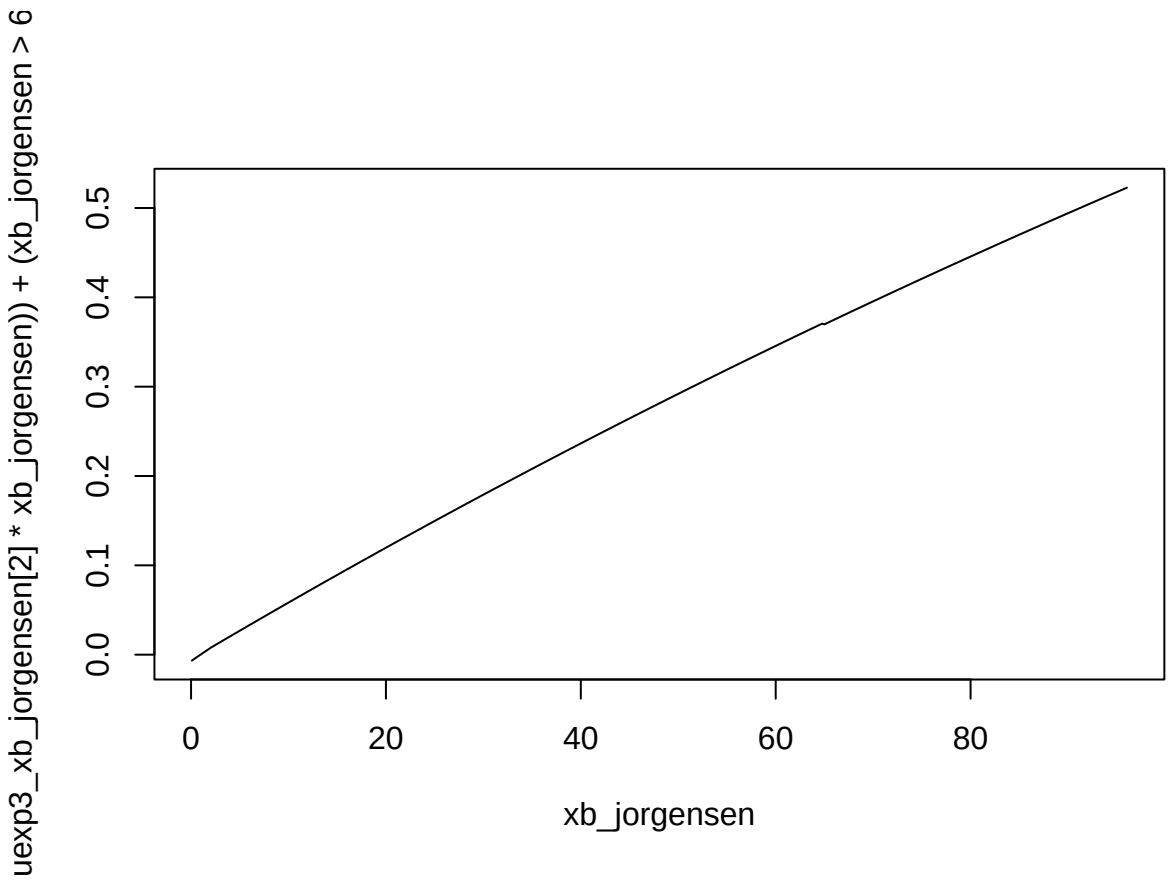
```
## $par
## [1] 0.001000000 0.003955892 0.002505107
##
## $value
## [1] 3.478313e-06
##
## $counts
## function gradient
##       50       50
##
## $convergence
## [1] 52
##
## $message
## [1] "ERROR: ABNORMAL_TERMINATION_IN_LNSRCH"
```




```r
stuexp3_xb_jorgensen
```

```
##      1.para      2.para      3.para
## 0.001000000 0.003955892 0.002505107
```

```r
plot(xb_jorgensen,
     ((xb_jorgensen > 0 & xb_jorgensen <= 2 ) * (1 - exp(-stuexp3_xb_jorgensen[1] * xb_jorgensen)) +
                                  (xb_jorgensen > 2 & xb_jorgensen <= 6 ) *
              (1 - exp(-stuexp3_xb_jorgensen[1] * 2))  +
                                  (exp(-2 * stuexp3_xb_jorgensen[2]) - exp(-stuexp3_xb_jorgensen[2] *
                                                                           xb_jorgensen)) +
                                  (xb_jorgensen > 6 & xb_jorgensen <= 65 ) *
              (1 - exp(-stuexp3_xb_jorgensen[1] * 2)) +
                                  (exp(-2 * stuexp3_xb_jorgensen[2]) - exp(-6 * stuexp3_xb_jorgensen[2]
                                  (exp(-6 * stuexp3_xb_jorgensen[3]) -
                                           exp(-stuexp3_xb_jorgensen[3] * xb_jorgensen))),
     type = "l")
```



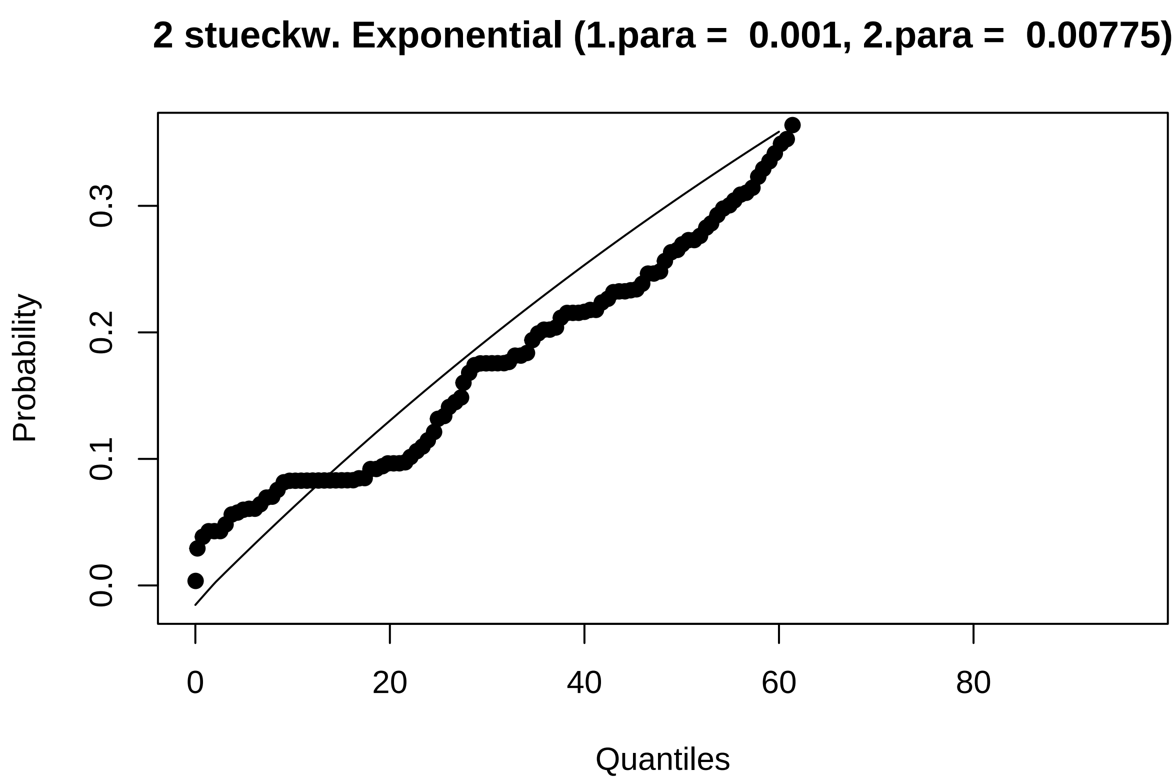
2-Stufige Exponetialverteilung

```r
# 2-Stufige Exponetialverteilung
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getstuexp2.R")

best_stuexp2_xr_jorgensen <- find_best_start_2parameter(p = yr_jorgensen/100,
                                                        q = xr_jorgensen,
                                                        max_beta = 1,
                                                        max_eta = 0.5,
                                                        steps_beta = 0.1,
                                                        steps_eta = 0.01,
                                                        fitting_function = getstuexp2)
```

```
## Bester Startparameter: 0 0.02
## Bester Fehler: 5.619892e-06
```

```
stuexp2_xr_jorgensen <- getstuexp2(
                            p = yr_jorgensen/100,
                            q = xr_jorgensen,
                            start = best_stuexp2_xr_jorgensen,
                            show.output = TRUE,
                            plot = TRUE,
                            wert1 = 2
                          )
```

```
## $par
## [1] 0.001000000 0.007752908
##
## $value
## [1] 5.619892e-06
##
## $counts
## function gradient
##        8        8
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

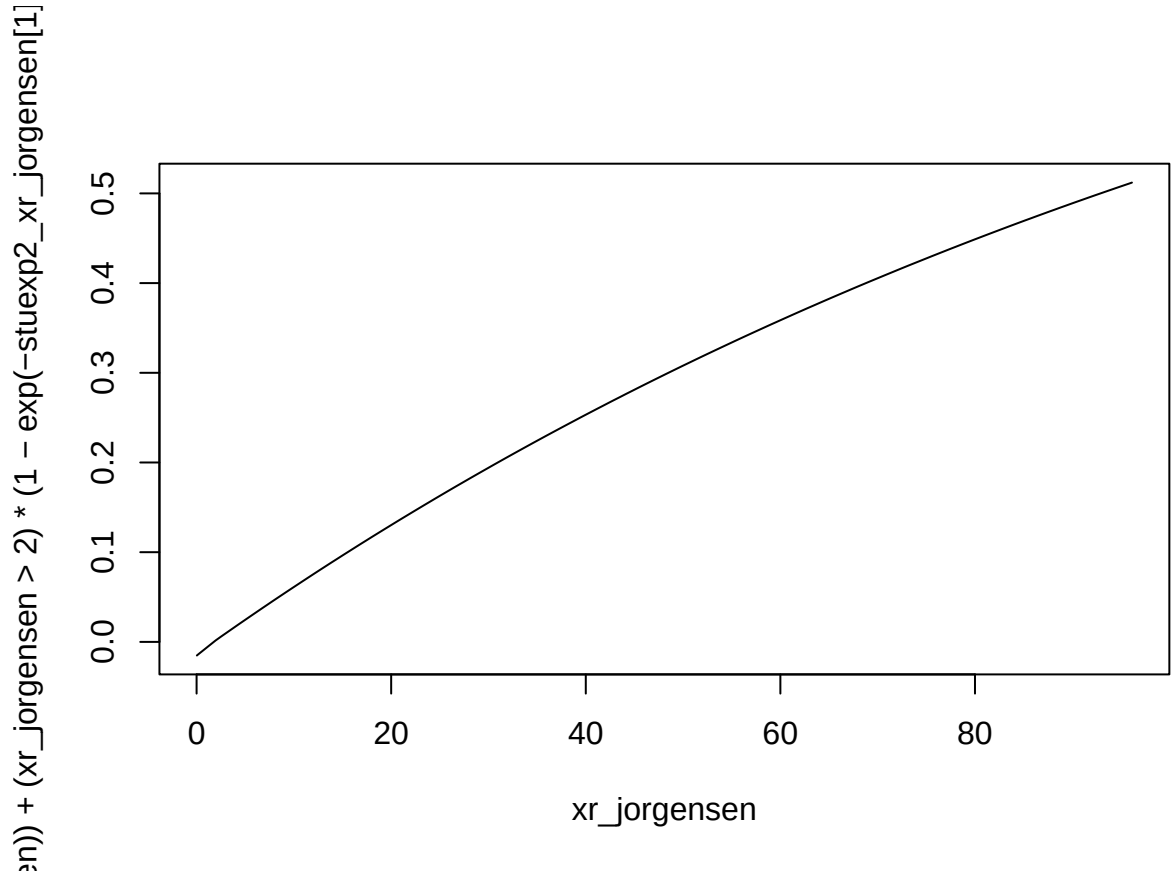


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
stuexp2_xr_jorgensen
```

```
##      1.para      2.para
## 0.001000000 0.007752908
```

```
plot(xr_jorgensen,
     ((xr_jorgensen > 0 & xr_jorgensen <= 2 ) * (1 - exp(-stuexp2_xr_jorgensen[1] * xr_jorgensen)) +
                                    (xr_jorgensen > 2 ) *  (1 - exp(-stuexp2_xr_jorgensen[1] * 2))  +
                                    (exp(-2 * stuexp2_xr_jorgensen[2]) -
                                             exp(-stuexp2_xr_jorgensen[2] * xr_jorgensen))),
     type = "l")
```



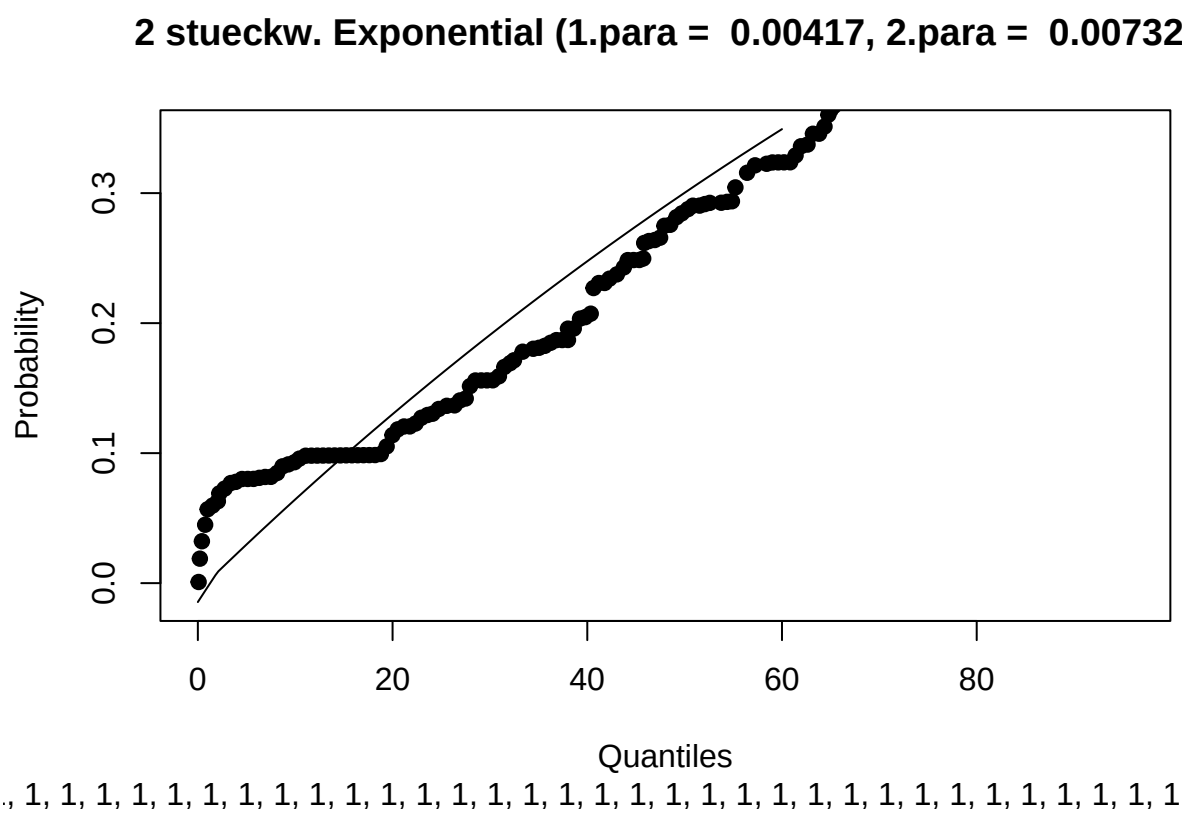

```
best_stuexp2_xb_jorgensen <- find_best_start_2parameter(p = yb_jorgensen/100,
                                                        q = xb_jorgensen,
                                                        max_beta = 1,
                                                        max_eta = 0.5,
                                                        steps_beta = 0.1,
                                                        steps_eta = 0.01,
                                                        fitting_function = getstuexp2)
```

```
## Bester Startparameter: 0.8 0
## Bester Fehler: 5.537071e-06
```

```
stuexp2_xb_jorgensen <- getstuexp2(
                        p = yb_jorgensen/100,
                        q = xb_jorgensen,
                        start = best_stuexp2_xb_jorgensen,
```

```
                        show.output = TRUE,
                        plot = TRUE,
                        wert1 = 2
                      )
```

```
## $par
## [1] 0.004172411 0.007319418
##
## $value
## [1] 5.537071e-06
##
## $counts
## function gradient
##       28       28
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

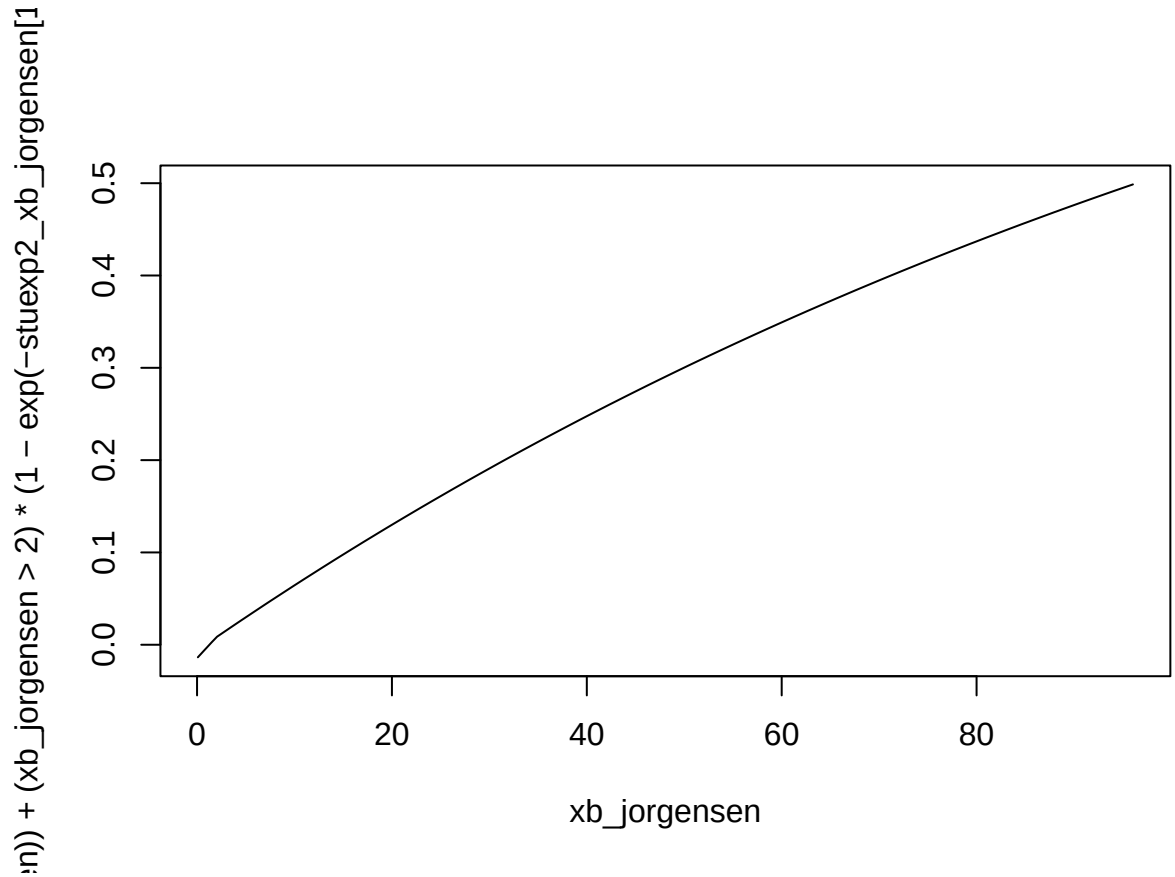


., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
stuexp2_xb_jorgensen
```

```
##      1.para      2.para
## 0.004172411 0.007319418
```

```r
plot(xb_jorgensen,
     ((xb_jorgensen > 0 & xb_jorgensen <= 2 ) * (1 - exp(-stuexp2_xb_jorgensen[1] * xb_jorgensen)) +
                                  (xb_jorgensen > 2 ) *  (1 - exp(-stuexp2_xb_jorgensen[1] * 2))  +
                                  (exp(-2 * stuexp2_xb_jorgensen[2]) -
                                          exp(-stuexp2_xb_jorgensen[2] * xb_jorgensen))),
     type = "l")
```



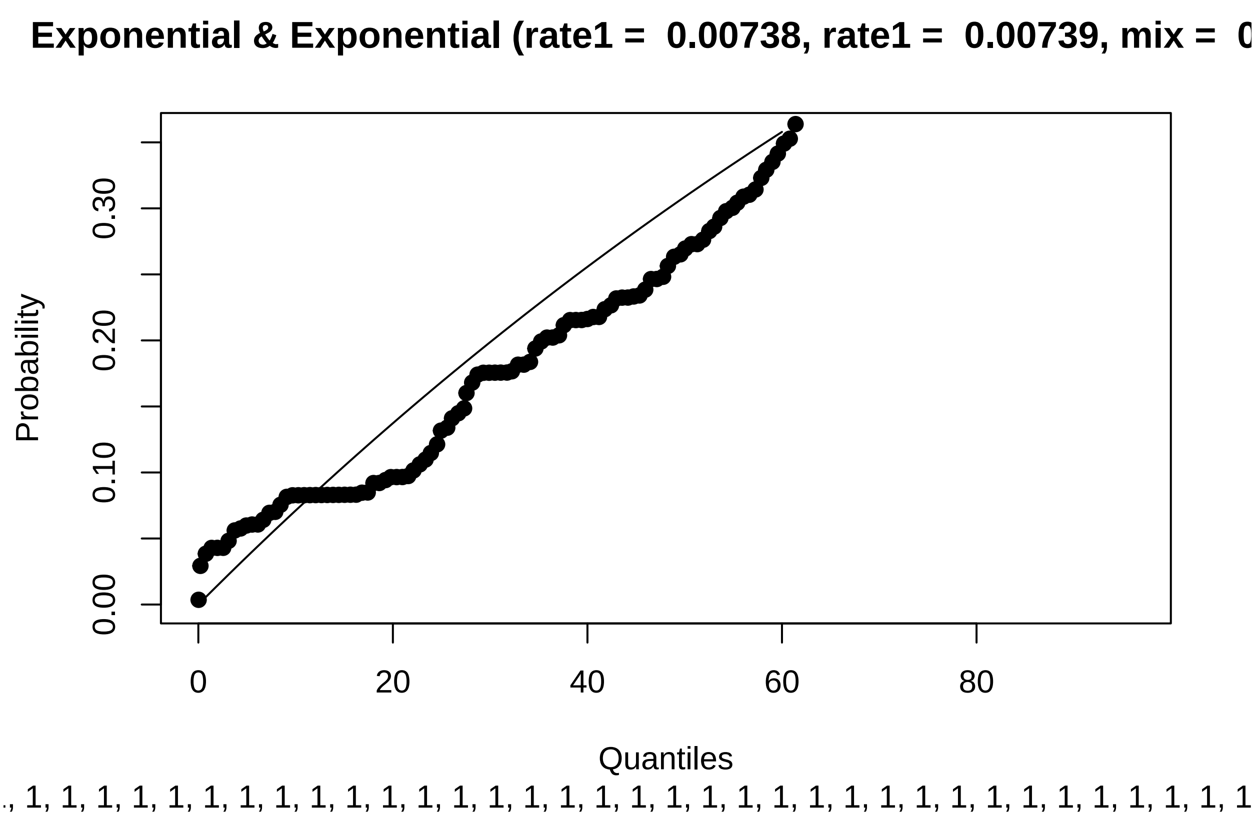
Mischung aus 2 Exponentialverteilungen

```r
# Mischung aus 2 Exponentialverteilungen
source("C:/Users/Chau/Documents/Masterarbeit/R Funktionen/getexex.R")

best_exex_xr_jorgensen <- find_best_start_3parameter(p = yr_jorgensen/100,
                                                     q = xr_jorgensen,
                                                     max_shape1 = 0.7,
                                                     max_shape2 = 0.7,
                                                     max_scale = 1,
                                                     steps_shape1 = 0.1,
                                                     steps_shape2 = 0.1,
                                                     steps_scale = 0.1,
                                                     fitting_function = getexex)
```

```
## Bester Startparameter: 0.6 0.3 0
## Bester Fehler: 6.120317e-06
```

```
exex_xr_jorgensen <- getexex(
                        p = yr_jorgensen/100,
                        q = xr_jorgensen,
                        start = best_exex_xr_jorgensen,
                        show.output = TRUE,
                        plot = TRUE
                      )
```

```
## $par
## [1] 0.007377371 0.007389922 0.001196593
##
## $value
## [1] 6.120317e-06
##
## $counts
## function gradient
##       47       47
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

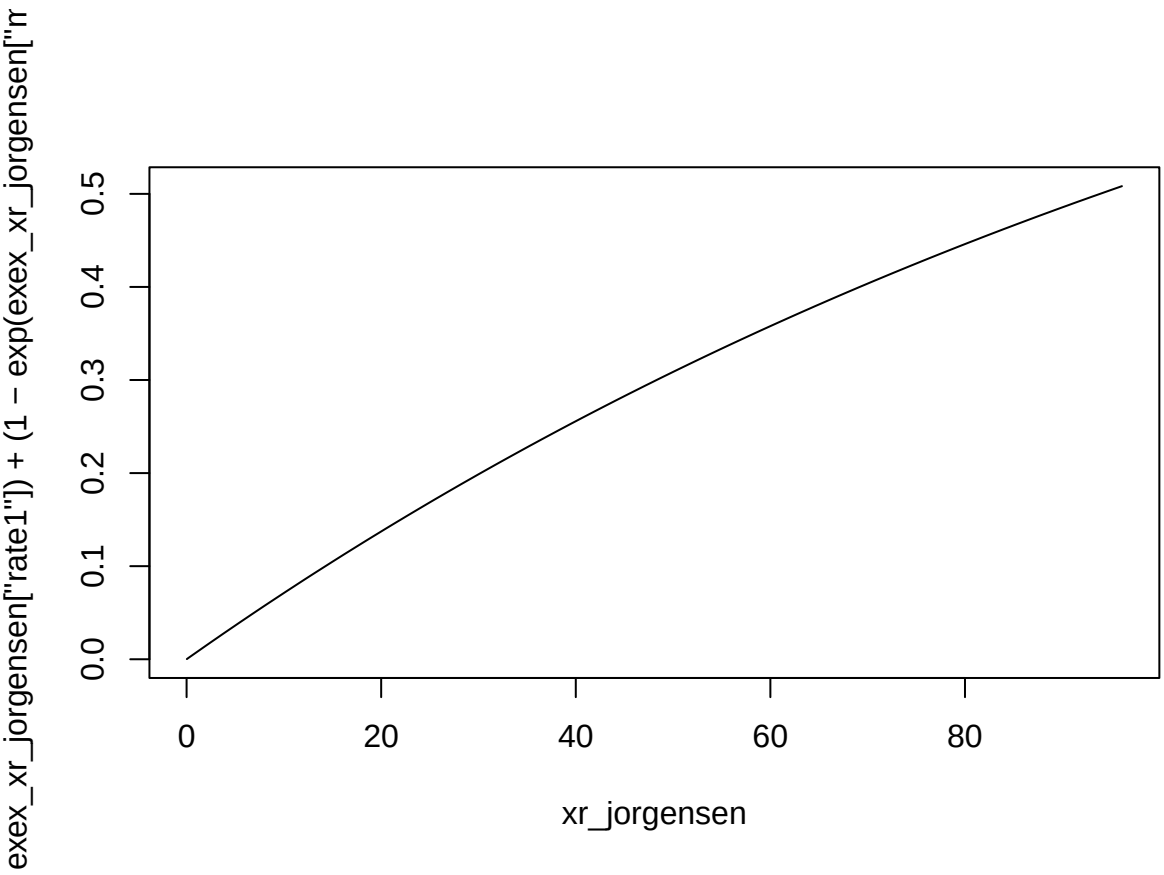



```
exex_xr_jorgensen
```

```
##        rate1        rate2          mix
```

```
## 0.007377371 0.007389922 0.001196593
```

```
plot(xr_jorgensen,
     (exp(exex_xr_jorgensen["mix"]) / ( 1 + exp(exex_xr_jorgensen["mix"])) *  stats::pexp(q = xr_jorgens
                                                                                rate = exex_xr_jorgensen["ra
                      (1 - exp(exex_xr_jorgensen["mix"]) / ( 1 + exp(exex_xr_jorgensen["mix"]))) *
              stats::pexp(q = xr_jorgensen,
                          rate = exex_xr_jorgensen["rate2"])),
     type = "l")
```



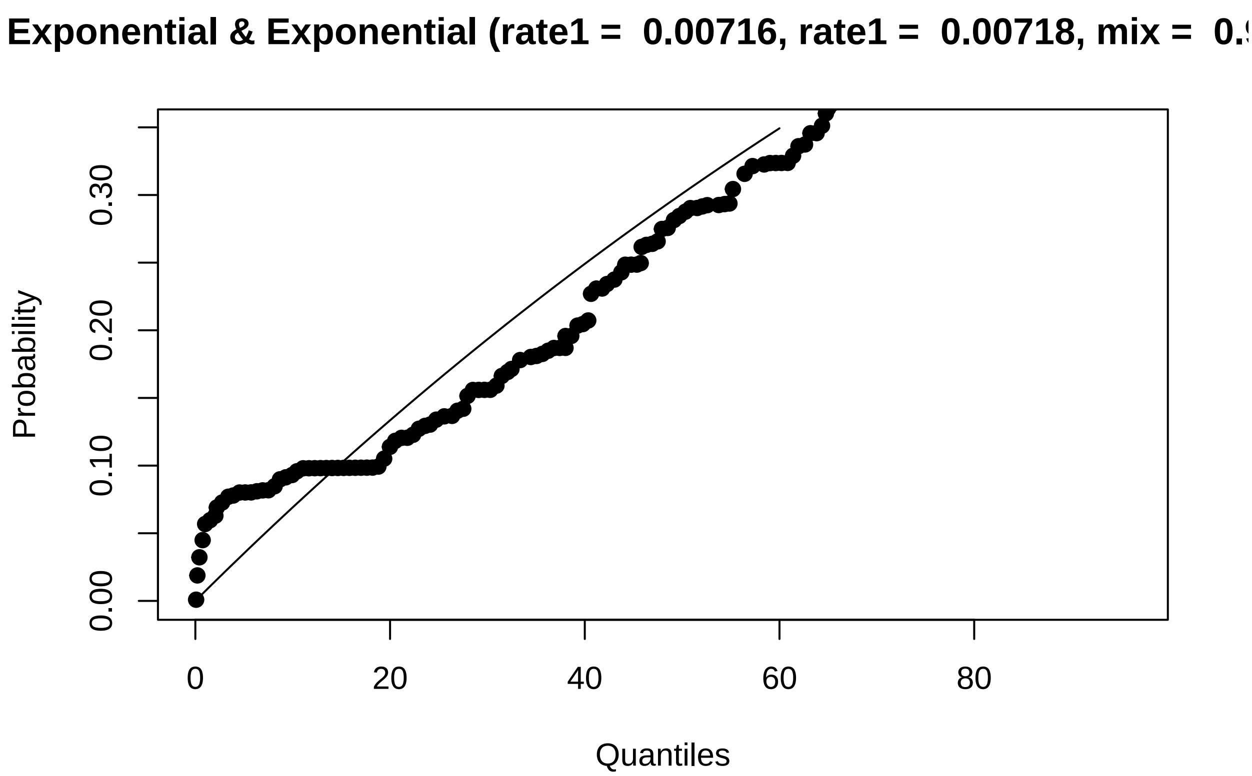

```
best_exex_xb_jorgensen <- find_best_start_3parameter(p = yb_jorgensen/100,
                                                     q = xb_jorgensen,
                                                     max_shape1 = 0.7,
                                                     max_shape2 = 0.7,
                                                     max_scale = 1,
                                                     steps_shape1 = 0.1,
                                                     steps_shape2 = 0.1,
                                                     steps_scale = 0.1,
                                                     fitting_function = getexex)
```

```
## Bester Startparameter: 0.4 0.6 0.4
## Bester Fehler: 5.456246e-06
```

```
exex_xb_jorgensen <- getexex(
                         p = yb_jorgensen/100,
                         q = xb_jorgensen,
                         start = best_exex_xb_jorgensen,
                         show.output = TRUE,
```

```
                        plot = TRUE
                      )
```

```
## $par
## [1] 0.007161282 0.007179031 3.972320815
##
## $value
## [1] 5.456246e-06
##
## $counts
## function gradient
##       36       36
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```



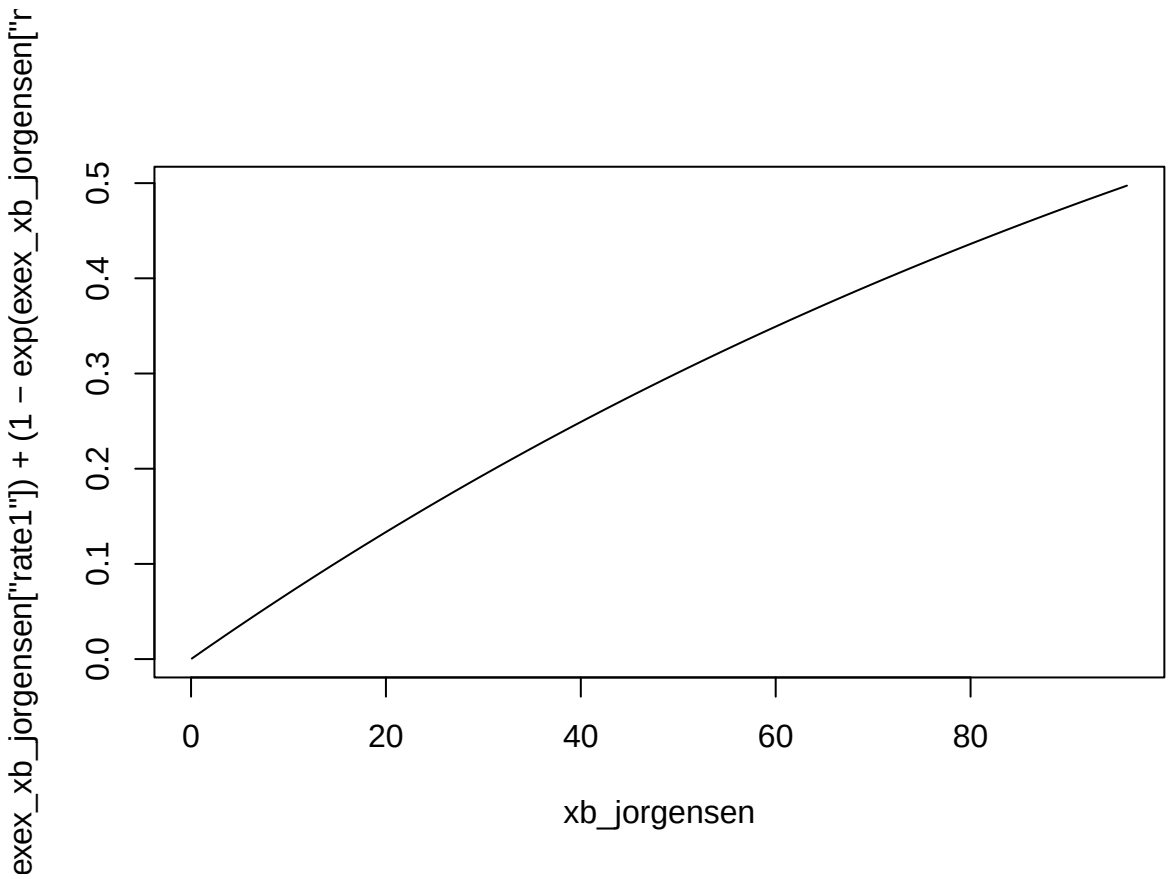
., 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1

```
exex_xb_jorgensen
```

```
##       rate1       rate2         mix
## 0.007161282 0.007179031 3.972320815
```

```
plot(xb_jorgensen,
     (exp(exex_xb_jorgensen["mix"]) / ( 1 + exp(exex_xb_jorgensen["mix"])) *  stats::pexp(q = xb_jorgens
                                                                                  rate = exex_xb_jorgensen["ra
```

```
                    (1 - exp(exex_xb_jorgensen["mix"]) / ( 1 + exp(exex_xb_jorgensen["mix"]))) *
            stats::pexp(q = xb_jorgensen,
                        rate = exex_xb_jorgensen["rate2"])),
    type = "l")
```



Ergebnnis

```
getvalue <- function(p, q, best_start, fitting_function){
  if(identical(as.character(substitute(fitting_function)), "getstuexp2")){
    output <- capture.output({
    result <- getstuexp2(p = p, q = q, start = best_start, show.output = TRUE,
                         plot = FALSE, wert1 = 2)
    })
  }
  else if(identical(as.character(substitute(fitting_function)), "getstuexp3")){
          output <- capture.output({
            result <- getstuexp3(
                          p = p, q = q, start = best_start,
                          show.output = TRUE, plot = FALSE, wert1 = 2, wert2 = 6)
          })
  }
  else{
    output <- capture.output({
    result <- fitting_function(p = p, q = q, start = best_start,
```

```
                                show.output = TRUE, plot = FALSE)
    })
  }

  # Berechne den Fehler (jorgensen_r_1$value) für die aktuellen Startparameter
  error <- as.numeric(gsub("\\[1\\]\\s+", "", output[5]))

  return(error)
}

best_test <- function(p, q, weibull, weib, weibex, weibex2, expweib, stuexp3, stuexp2,
                      exex, start_weibull, start_weib, start_weibex, start_weibex2,
                      start_expweib, start_stuexp3, start_stuexp2, start_exex,
                      group){
  weibull_val <- getvalue(p, q, start_weibull, getweibullpar)
  weib_val <- getvalue(p, q, start_weib, getweibpar)
  weibex_val <- getvalue(p, q, start_weibex, getweibex)
  weibex2_val <- getvalue(p, q, start_weibex2, get2weibex)
  expweib_val <- getvalue(p, q, start_expweib, getexpweib)
  stuexp3_val <- getvalue(p, q, start_stuexp3, getstuexp3)
  stuexp2_val <- getvalue(p, q, start_stuexp2, getstuexp2)
  exex_val <- getvalue(p, q, start_exex, getexex)

  error_distribution_pairs <- list(
    list(weibull_val, "W"),
    list(weib_val, "e.W."),
    list(weibex_val, "M. W&E"),
    list(weibex2_val, "M. e.W&E"),
    list(expweib_val, "e.W o. lambda = 1"),
    list(stuexp3_val, "3 s.E."),
    list(stuexp2_val, "2 s.E."),
    list(exex_val, "M. E&E")
  )

  # Suchen Verteilung mit dem kleinsten Fehler
  best_pair <- error_distribution_pairs[[which.min(sapply(
    error_distribution_pairs, function(pair) pair[[1]]))]]

  # Drucken Sie die Ergebnisse
  cat("Beste Verteilung:", best_pair[[2]], "\n")
  cat("Bester Fehler:", best_pair[[1]], "\n")
  cat("Gruppe: ", group)

  return(c(group, best_pair[[2]],  best_pair[[1]]))
}

best_savr <- best_test(yb_jorgensen/100, xb_jorgensen, jorgensen_b_1, weibbull_xb_jorgensen,
                       weibex_xb_jorgensen, weibex2_xb_jorgensen, expweib_xb_jorgensen,
                       stuexp3_xb_jorgensen, stuexp2_xb_jorgensen, exex_xb_jorgensen,
                       best_jorgensen_b_1, best_weibbull_xb_jorgensen,
                       best_weibex_xb_jorgensen, best_weibex2_xb_jorgensen,
                       best_expweib_xb_jorgensen, best_stuexp3_xb_jorgensen,
                       best_stuexp2_xb_jorgensen, best_exex_xb_jorgensen, "SAVR")
```

```
## Beste Verteilung: M. W&E
## Bester Fehler: 5.537916e-07
## Gruppe:  SAVR
```

```
best_tavr <- best_test(yr_jorgensen/100, xr_jorgensen, jorgensen_r_1, weibbull_xr_jorgensen,
                       weibex_xr_jorgensen, weibex2_xr_jorgensen, expweib_xr_jorgensen,
                       stuexp3_xr_jorgensen, stuexp2_xr_jorgensen, exex_xr_jorgensen,
                       best_jorgensen_r_1, best_weibbull_xr_jorgensen,
                       best_weibex_xr_jorgensen, best_weibex2_xr_jorgensen,
                       best_expweib_xr_jorgensen, best_stuexp3_xr_jorgensen,
                       best_stuexp2_xr_jorgensen, best_exex_xr_jorgensen, "TAVR")
```

```
## Beste Verteilung: M. e.W&E
## Bester Fehler: 8.776434e-07
## Gruppe:  TAVR
```

```
tab <- matrix(c("NOTION", "alle", "TSM", best_tavr[1], best_tavr[2],
                best_tavr[3], weibex2_xr_jorgensen[1:3], NA, NA, NA,
                weibex2_xr_jorgensen[4],
                "NOTION", "alle", "TSM", best_tavr[1], "M. W&E",
                getvalue(yr_jorgensen/100, xr_jorgensen, best_weibex_xr_jorgensen, getweibex),
                NA, NA, weibex_xr_jorgensen[1:2], NA, weibex_xr_jorgensen[3:4],
                "NOTION", "alle", "TSM",  best_savr[1], best_savr[2],
                best_savr[3], NA, NA, weibex_xb_jorgensen[1:2], NA,
                weibex_xb_jorgensen[3:4]),
              ncol=13, byrow=TRUE)

rownames(tab) <- NULL
colnames(tab) <- c('Studie', 'PG', 'EP', 'GR', 'Verteilung', 'SSE', '$\\alpha$',
                   '$\\theta$', '$\\lambda_1$', '$\\lambda_2$', '$\\lambda_3$',
                   '$\\vartheta$', '$\\psi$')

results <- as.data.frame(tab)

# Speichern
write.table(results, "results_jorgensen.txt", sep = "\t", row.names = FALSE)

# Funktion zur Überprüfung von NA-Werten für Zeichenketten und numerische Werte
is_non_empty <- function(x) {
  return(!is.na(x) & x != "")
}

# Spalten mit mindestens einem nicht-NA-Wert ermitteln
nicht_leere_spalten <- colSums(sapply(results, is_non_empty)) > 0

# Konvertieren Sie die Tabelle in eine Markdown-Tabelle
print(results[, nicht_leere_spalten])
```

```
##   Studie   PG  EP   GR Verteilung          SSE         $\\alpha$
## 1 NOTION alle TSM TAVR   M. e.W&E 8.776434e-07 0.318085638347565
## 2 NOTION alle TSM TAVR     M. W&E 9.427333e-07              <NA>
## 3 NOTION alle TSM SAVR     M. W&E 5.537916e-07              <NA>
##          $\\theta$      $\\lambda_1$     $\\lambda_2$     $\\vartheta$
## 1 47.5324992810653 0.313042930144028             <NA>             <NA>
## 2             <NA>  1.55865720109492 112.219015159449 3.78019047741798
```

```
## 3             <NA>   1.6887982737076 116.055116397409 1.34734657059735
##            $\\psi$
## 1 2.43780761004889
## 2 3.02502395742585
## 3 2.56340038363206
```